Copy No. _____

Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

# DSSP50 Build 090910 Documentation

## *Part 2: Input Reference*

*M. Mackay*

**Defence R&D Canada – Atlantic**

Technical Memorandum
DRDC Atlantic TM 2009-227
December 2009

Canada

This page intentionally left blank.

# DSSP50 Build 090910 Documentation, Part 2: Input Reference

M. Mackay
Defence R&D Canada – Atlantic

**Defence R&D Canada – Atlantic**
Technical Memorandum
DRDC Atlantic TM 2009–227
December 2009

Author

*Original signed by M. Mackay*

M. Mackay
Defence Scientist

Approved by

*Original signed by N. Pegg*

N. Pegg
Head, Warship Performance

Approved for release by

*Original signed by R. Kuwahara for*

C. Hyatt
Chair, Document Review Committee

© Her Majesty the Queen as represented by the Minister of National Defence, 2009
© Sa majesté la reine, représentée par le ministre de la défense nationale, 2009

## Abstract

DSSP50 is a computer code for evaluating the dynamics and manoeuverability of submarines and smaller streamlined underwater vehicles. This document is the input reference for the program. It outlines input structure, and details the syntax and defaults for every input record. Other documents overview the program and describe the underlying algorithms.

## Résumé

DSSP50 est un code machine permettant d'évaluer la dynamique et la manœuvrabilité des sous-marins et des plus petits véhicules sous-marins hydrodynamiques. Le présent document est une référence d'entrée du logiciel. Il indique la structure d'entrée et donne les détails de la syntaxe et des défauts pour chaque dossier d'entrée. D'autres documents font le survol du logiciel et décrivent les algorithmes sous-jacents.

This page intentionally left blank.

## Executive Summary

**DSSP50 Build 090910 Documentation, Part 2: Input Reference**
**M. Mackay; DRDC Atlantic TM 2009–227; Defence R&D Canada – Atlantic; December 2009.**

**Introduction:** DRDC Submarine Simulation Program, version 5.0 (DSSP50), was developed to predict the manoeuverability of submarines and smaller underwater vehicles. It combines the capability of two earlier codes, DSSP21 and DSSP40. The program is used to calculate loads on captive models, derive hydrodynamic coefficients, simulate tactical manoeuvers, and generate Manoeuvering Limitation Diagram (MLD) curves.

This document is the input reference to DSSP50. It outlines preparation of input for the definition of the vehicle, its systems, and the hydrodynamic and manoeuvering simulations to be done, and details the syntax and defaults for each input record.

**Significance:** The target user group for the program is DRDC and other Government researchers, CF engineers, and CF contractors.

DSSP50 was formulated to balance accuracy, which is sufficient for predicting complex tactical manoeuvers, with computational speed. Speed is required for systematic studies in support of preliminary design and for MLD calculations. Making it multifunctional ensured that all calculations and simulations are based on a common set of algorithms, and thus consistent.

Although input to the program initially appears complex because of a large number of options, experience with its predecessors suggests that it should be relatively easy to learn. This reference is for both new and experienced users.

**Future Development:** Future development of the program may include such features as near-bottom operation.

## Sommaire

**DSSP50 Build 090910 Documentation, Part 2: Input Reference**
**M. Mackay; DRDC Atlantic TM 2009–227; R & D pour la défense Canada – Atlantique; décembre 2009.**

**Introduction:** Le logiciel de simulation sous-marine de RDDC, version 5.0 (DSSP50), a été développé afin de prévoir la manœuvrabilité des sous-marins et des véhicules sous-marins plus petits. Il combine la capacité de deux codes antérieurs, DSSP21 et DSSP40. Le logiciel sert au calcul des charges sur des maquettes entravées, à calculer les coefficients hydrodynamiques, à simuler les manœuvres tactiques et à générer des courbes des schémas de limitation des manœuvres (SLM).

Le présent document est la référence d'entrée au DSSP50. Il indique la préparation d'entrée pour la définition du véhicule, ses systèmes et les simulations hydrodynamiques et de manœuvrabilité à effectuer et donne les détails de la syntaxe et des défauts pour chaque dossier d'entrée.

**Portée:** Le groupe utilisateur cible pour le logiciel est RDDC et d'autres chercheurs du gouvernement, les ingénieurs des FC et les entrepreneurs des FC.

Le DSSP50 a été formulé pour équilibrer la précision, qui est suffisante pour prévoir des manœuvres tactiques complexes, à la vitesse computationnelle. La vitesse est requise pour des études systématiques de soutien à la conception préliminaire et aux calculs des SLM. Le fait de le rendre multifonctionnel a permis de garantir que tous les calculs et les simulations sont basés sur un ensemble commun d'algorithmes et sont ainsi compatibles.

Quoique l'entrée dans le logiciel semble initialement complexe compte tenu du grand nombre d'options, l'expérience avec ses prédécesseurs donne à penser qu'elle devrait être relativement facile à apprendre. La présente référence est à la fois pour les nouveaux utilisateurs et ceux d'expérience.

**Recherches futures:** Les recherches futures pourraient comprendre des caractéristiques comme les opérations près du fond.

## Table of Contents


Abstract ..... i
Résumé ..... i
Executive Summary ..... iii
Sommaire ..... iv
Table of Contents ..... v
List of Figures ..... viii
1 Introduction ..... 1
2 General Considerations ..... 2
2.1 Input Syntax ..... 2
2.2 Axis Systems ..... 4
2.3 Applicability of the Program ..... 6
3 Common Input Records ..... 7
3.1 Component Weights ..... 7
3.2 Label ..... 7
3.3 Pop ..... 8
3.4 Simulation Input Records ..... 8
4 Root.geo Level 1 Input ..... 10
4.1 AddedMass and Inertial Properties ..... 11
4.2 Component Specification ..... 13
4.3 Propulsion ..... 14
4.4 The Hydrodynamic Reference ..... 15
4.5 Submarine Systems ..... 16
4.6 Other Input ..... 18
5 Root.geo Level 2 Hull Input ..... 19
5.1 Hull Geometry ..... 21
5.2 Additional Parameters ..... 24
6 Root.geo Level 2 Lift Input ..... 26
6.1 Appendage geometry ..... 30
6.2 Appendage Type ..... 31
6.3 Control by Appendage Deflections ..... 33
6.4 Other Parameters ..... 37
6.5 Copying Appendage Specifications ..... 38
7 Root.geo Level 2 Lump Input ..... 39
7.1 Lump Spatial and Inertial Properties ..... 40

7.2 Lump Coefficient Model ..... 41
7.3 Plane Deflection Control ..... 42
7.4 Other Records ..... 44
8 Root.geo Level 2 OOPCalc Input ..... 46
9 Root.geo Level 2 Propulsor Input ..... 48
9.1 Propulsor Geometry ..... 49
9.2 Other Propulsor Parameters ..... 50
9.3 Saving and Copying ..... 51
10 Root.geo Level 2 WagBInit Input ..... 52
11 Root.geo Level 2 Autopilot Input ..... 54
11.1 Deriving Control Signals ..... 55
11.2 AutoDepth Level 2 Input ..... 56
11.3 AutoHead Level 2 Input ..... 57
11.4 AutoProp Level 2 Input ..... 57
11.5 AutoRoll Level 2 Input ..... 58
12 Root.geo Level 2 Compress Input ..... 59
13 Root.geo Level 2 HPAir and MBT Input ..... 60
13.1 HPAir Level 2 Input ..... 62
13.2 MBT Level 2 Input ..... 63
14 Root.geo Level 2 WTC Input ..... 64
15 Root.geo Level 2 Comp, Trim, and TrimComp Input ..... 66
16 Root.geo Level 2 Plot Input ..... 67
17 Root.sim Control Deflection Commands ..... 71
17.1 Direct Deflection Input ..... 72
17.2 Quasi-Direct Deflection Input ..... 73
17.3 Indirect Deflection Input ..... 74
17.4 Global Deflection Input ..... 75
17.5 Conflicting Deflection Commands ..... 75
18 Root.sim Heading Specification ..... 76
19 Root.sim MBT Blowing and Venting Commands ..... 78
20 Root.sim WTC Flooding Commands ..... 80
21 Root.sim WTC Trim and Compensation Commands ..... 81
22 Root.sim Level 1 Input ..... 83
22.1 Calculation and Simulation Specification ..... 83
22.2 Other Input ..... 85
23 Root.sim Level 2 Static Input ..... 86
23.1 Non-Translational Input ..... 86
23.2 Direct Translational Input ..... 87

23.3 Indirect Translational Input ..... 88
23.4 The Loop Record ..... 89
24 Root.sim Level 2 Coefficnt Input ..... 90
25 Root.sim Level 2 Captive Input ..... 93
26 Root.sim Level 2 Free Input ..... 97
26.1 Free Simulation parameters ..... 100
26.2 Command Records ..... 103
27 Free and MLD Initialize Input ..... 108
27.1 Lump Control Level 4 Initialization Input ..... 112
28 Free and MLD Settings Input ..... 113
29 Root.sim Level 2 MLD Input ..... 116
29.1 MLD Level 3 Flood Input ..... 120
29.2 MLD Level 3 Jam Input ..... 125
30 DSSP50.myp Input ..... 129
31 Concluding Remarks ..... 131
References ..... 133
Annex A. Dictionary Lists ..... 135
Annex B. Hydrodynamic Coefficent Equivalence Table ..... 137
Nomenclature ..... 143
Distribution ..... 145
Distribution ..... 145

## List of Figures


| | | |
|---|---|---|
| Figure 1. | Principal earth-fixed and boat-fixed axis systems | 5 |
| Figure 2. | Target trajectory and reference axes relative to chart axes | 6 |
| Figure 3. | Hull definition nomenclature | 20 |
| Figure 4. | Appendage definition nomenclature | 27 |
| Figure 5. | Hull/sail input nomenclature | 27 |
| Figure 6. | Endplate input nomenclature | 28 |
| Figure 7. | Flapped appendage input nomenclature | 28 |
| Figure 8. | Schematic of three bottle groups and three ballast tanks | 61 |
| Figure 9. | `AcroSpin` plot screen shot | 68 |
| Figure 10. | `Maple` plot screen shot | 68 |
| Figure 11. | `MatLab` plot screen shot using the keyword `Colour` | 69 |
| Figure 12. | `MatLab` plot screen shot using the keyword `White` | 70 |
| Figure 13. | `VRML` plot from Myriad Viewer; wire frame rendering | 70 |
| Figure 14. | `VRML` plot from Myriad Viewer; shaded rendering | 71 |
| Figure 15. | A variation of figure 8 (ballast blowing) | 79 |
| Figure 16. | Schematic of trim and compensation system | 81 |

# 1 Introduction

Computer program DSSP50 evaluates the hydrodynamics and manoeuverability of underwater vehicles. It is primarily for submarines although it can be applied smaller streamlined submersibles. Hydrodynamic loads on the vehicle are calculated with a mix of fast semiempirical and coefficient based methods. Speed is necessary for systematic parameter variation during preliminary design, for evaluating emergency recovery procedures, and for other applications requiring many simulation runs.

DSSP50 documentation is in three parts:

- A general program guide, reference [1], primarily for new users.
- An input reference, this memorandum, for all users.
- A description of the algorithms, reference [2], for background information.

After some familiarity with the program is acquired, the principal resource will be this memorandum.

The main function of this document is to facilitate preparation of the input files. Minimum input requirements are fairly simple, but optional parameters are available to accommodate a wide range of vehicle configurations and to fine-tune the calculations. The hydrodynamic models and other aspects of the program are described elsewhere and summarized in reference [2]; they are discussed here in only sufficient detail as required to define the input.

New users are advised to read this reference with particular attention to section 2, General Considerations. They should also do the tutorial in reference [1] and review reference [2]. More experienced users will only need to refer to the input record descriptions as required. Input keywords are indexed at the end of this document.

The normal release of DSSP50 comprises a Windows executable **dssp50.exe** with the run-time library **salflibc.dll** and some other files including those created by the author for the tutorial.

Developmental status of the code is indicated by a build number consisting of the release date in *yymmdd* format. (In output from the program, an additional character may be appended: **A**, **B**, or **X**, indicating that the build is considered to be a preliminary, limited, or experimental release respectively. After sufficient validation the qualifier will be dropped.) The build at time of writing is 090910; subsequent builds will be accompanied by updates and additions to the documentation as required.

The program proceeds in two phases:

- Vehicle model and systems setup. The user must provide a file defining the vehicle geometry or hydrodynamic coefficients, or a combination of the two, together with basic systems information. The data is extensively checked and as much preprocessing of it is done as possible.
- Hydrodynamic and manoeuvering simulations. The simulations to be run are defined in sequence on an optional input file. Simulations include static and dynamic

loads on a captive vehicle, hydrodynamic coefficient generation, free vehicle manoeuvering, and Manoeuvering Limitation Diagram (MLD) curve calculation. If the simulation input file is absent, there is no work done in the second phase.

The objective in making the program multifunctional was to base all calculations and simulations on a common set of hydrodynamic modeling subroutines. The underlying hydrodynamic models are therefore consistent, and improvements and enhancements to the models will be applied universally.

DSSP50 is programmed in Fortran. Some parts of it are from much earlier programs, other parts are quite new. The code was revised to compile under Fortran 95 syntax rules, but only a handful of features are unlikely to compile under Fortran 77. One feature, the library routine required to extract the optional command line parameter, is compiler-dependent. DSSP50 was developed on a Microsoft Windows PC using the Salford FTN95 Win32 version 2.00 compiler, and checked with STI's Understand for Fortran, version 1.4. Although the code is virtually platform-independent, the documentation assumes a Windows platform where relevant.

The following conventions in the documentation should be noted. Object names (program units, etc.) and DSSP50 input keywords are in a capitalized typewriter font. Higher level keywords following a lower level one are indicated by an arrow: `Lower` → `Higher`. The words 'simulation' and 'run' are generally synonymous except for an MLD simulation which comprises a number of individual runs.

In this reference, input records are summarized in the following manner:

| `Keyword` *argument 1*, [*argument 2*] | | | | → |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *argument 1* | description | | | value |
| *argument 2* | description | units | limits | value |

Optional arguments are enclosed in brackets: [ ]. Units, range, and default values are given if applicable. The default applies to the absence of an optional argument or to the absence of the record altogether. An arrow in the top right corner indicates that there should be higher level input to follow.

# 2 General Considerations

## 2.1 Input Syntax

Input files *Root*`.geo` and *Root*`.sim` are prepared by the user with a text editor. The input character set for DSSP50 comprises upper and lower case letters, the space, digits 0 to 9, the 'printable' characters `*()-=+;:',<.>/?'~!@#$%^&_\|[{]}"`, and the tab. Unrecognized characters generally result in a fatal error. Most input is not case sensitive.

In order to allow these files to be more readable, a simple input parsing scheme is employed. Each line in the input file is treated as a separate record of up to 71 characters consisting of words separated by one or more delimiters and terminated by an optional end-of-record character. The word delimiters are the space, comma ',', semicolon ';', and tab. The end-of-record (EOR) characters are the double quote '"', exclamation point '!', and percentage sign '%'. A word is a string of any other characters. Input following an EOR character is ignored and therefore functions as a comment. Any input that is encountered after the input requirements of a record are fulfilled is similarly ignored.

Input is controlled by keywords. A keyword consists of a string of up to ten characters contained in one of the parser's dictionaries. Listings are given in annexes A and B, and are indexed at end of this report. Most input records consist of a keyword followed by one or more arguments. An argument may be numerical, some non-keyword text, or another keyword. Since some arguments are optional, it is prudent to always precede a following comment by an EOR character.

Unless noted, keywords discussed in this guide are contained in the main dictionary. They may be abbreviated by truncation so long as the result is not ambiguous. For example, keywords `Left` and `Length` can be abbreviated by `Lef` and `Len` respectively, but `Le` is ambiguous. On the other hand, an unabbreviated keyword is distinguished from any abbreviations that may be identical: `Delta` and `DeltaLim` are therefore recognized as separate keywords. For this reason, keyword abbreviation should be used with care since the result may be unexpected. Finally, it should be noted that the same keyword may be used for different purposes in different parts of the input, its interpretation depending on the context.

With keyword controlled input, records can in principle appear in any order in the input file. However, in `DSSP50` this freedom is limited to within an input 'level': input is initially at Level 1 and is switched to Level 2 and higher by specific keywords. Input order is obviously important within a data block, i.e., a sequence of physical records that is treated as a single logical record. And some simulations, notably for static load estimation, execute records in the *Root*`.sim` file in input order, so that changing that order may affect the results.

Text records provide indentification and other information for printed output. They consist of the keyword `Text`, one or more delimiters, and a string of any characters in the `DSSP50` character set. On output, case in a `Text` string is preserved.

A label string uniquely identifies a vehicle component. It consists of 1 to 24 alphabetic characters preceded by a label character, '#' or '?'. Labels are useful for clarifying the object of control commands, and may be defined by the user; they default to `#HULL001`, `#LIFT001`, `#PROP001`, etc. Labels are not case sensitive on input.

Numerical values in input are evaluated by specific parsing routines. An integer can have no more than 20 characters, and is read, right justifed, in Fortran `I20` format. A floating-point value can have no more than 30 characters. Initially it is read, right

justified, in I30 format and converted to a double precision variable; if that fails, it is re-read in F30.0 format into a double precision variable.

Blank lines in input are ignored, and purely comment lines should start with an EOR character otherwise the parser will try to interpret the first delimited string as a keyword.

## 2.2 Axis Systems

The relationship between earth-fixed and body-fixed axis systems is discussed in standard textbooks, e.g., Etkin [3]. There are two principal body-fixed axis systems in DSSP50: vehicle axes, in which vehicle geometry and most other input is defined, and reference axes, in which output quantities such as hydrodynamic forces and moments are defined. Both systems are righthanded. Vehicle axes are determined indirectly by the $(x, y, z)$ coordinates of vehicle geometry which are initially defined in file *Root*.geo.

Vehicle components use vehicle axes for their definition. Coordinates and lengths are in metres, and angles are in degrees. Coefficient and hybrid models use the concept of a lump as the entity with which hydrodynamic coefficients are associated, and lumps are also located in vehicle axes.

Reference axes are generally defined by the user (for a single-hulled component model they default to vehicle axes shifted to an origin at the hull CB). To be most meaningful, reference axes should coincide with conventional hydrodynamic axes, i.e., with $x$ directed forward, $y$ to starboard, and $z$ downward. Likewise, but also for computational stability in a free manoeuver, the reference axis origin should be in the vicinity of the vehicle CB and CG.

The reference origin trajectory is always output in a free manouever, but the user may define an additional point, called the target, which is also tracked. This is typically a point of interest such as the attachment point for a towed array. Defining it as the target avoids numerical difficulties that could be encountered when shifting the reference origin to achieve the same result. (The problem is principally with the matrix comprising the inertial plus added mass properties of the submarine; it must be inverted to decouple the equations of motion for integration in time. Since internal DSSP50 calculations are dimensioned, some elements of the matrix grow rapidly as the reference origin is moved away from the principal axes of the vehicle, making it ill-conditioned for inversion. Nondimensionalizing this part of the calculations may alleviate the problem, but that is outside the scope of current code development.)

In a free-running manoeuver there is a set of earth-fixed axes $(x_0, y_0, z_0)$ that are parallel to the reference axes at the start of the simulation with their origin on the surface rather than at initial depth $z_{0\bullet}$. Consequently, $(x_0, y_0, z_0)_{\text{start}} = (0, 0, z_{0\bullet})$, where here, as elsewhere in this document, subscript $\bullet$ indicates an initial condition. The trajectory of the reference origin in $(x_0, y_0, z_0)$ coordinates is tabulated on file simnnn.txt, which includes the state variables $u$, $v$, $w$, $p$, $q$, and $r$ in the reference coordinate system.

Internally, the program uses a separate local axis system for each component, including lumps. For a hull component the origin is at its longitudinal CB, local $x$ is directed forward along the longitudinal axis, and $y$ and $z$ are determined from vehicle axes or are user-defined. For a lifting appendage the origin is at the root mid-chord, local $y$ is outward along the mid-chord line, $z$ is normal to the effective plane of the appendage, and $x$ is obtained from the cross product of $y$ and $z$. In a few instances, printed output is in local axes. The main reason for the user to be aware of them is that individual control deflections will be defined in appendage local axes.

These axis systems are sketched in figure 1 for a component model with the vehicle axis origin at the bow of the submarine. For all-moving rudders on the same shaft, the definition of appendage local axes results in deflections of opposite sign on the upper and lower rudders; the same observation applies to other pairs of control planes.

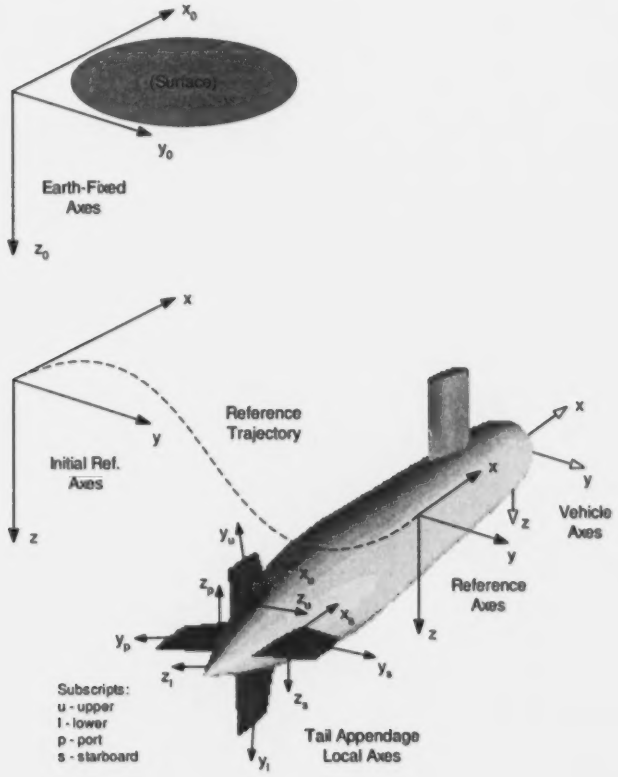


**Figure 1.** Principal earth-fixed and boat-fixed axis systems for a component model.

There is another earth-fixed axis system to note in connection with free-running manoeuvers: chart axes $(x_e, y_e, z_e)$. In this system $x_e$ is directed North, $y_e$ East, and $z_e$ down. At present, they serve no purpose other than to locate the target point of the vehicle on a notional chart, but may be useful for future developments such as defining density as a spatial variable.

In chart axes, the target initial position and heading are defined by `Origin` and `Heading` records (section 26.1) as illustrated in figure 2. Target coordinates $(x_T, y_T, z_T)$ in chart axes are output on files *Root*`.spr` and `aux`*nnn*`.txt`; the latter also tabulates target translational velocities $u_T$, $v_T$, and $w_T$ in reference axes.

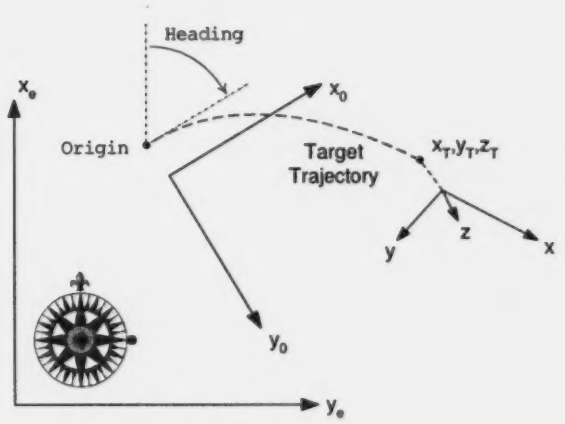


**Figure 2.** The target trajectory, $(x_0, y_0, z_0)$ axes, and reference axes $(x, y, z)$ relative to the chart axis system.

## 2.3 Applicability of the Program

It is implicit in many algorithms used in the component model of `DSSP50` [2] that the vehicle is streamined and at moderate incidence, i.e., the components are reasonably slender and have their hydrodynamic axis — the longitudinal axis of a hull or the chordline of a lifting appendage — oriented more-or-less in the direction of motion (or against the direction of incident flow). The component model therefore cannot adequately represent bluff or irregular underwater vehicles such as ROVs, but can model submarines, many UUVs, and towfish.

The coefficient model [2], on the other hand, can represent any vehicle so long as there is a sufficient number of relevent coefficients, whose values are known with sufficient accuracy, to calculate the hydrodynamic loads on it.

# 3 Common Input Records

The DSSP50 file system is outlined in reference [1]. There are a number of input records that appear in several places in files *Root*`.geo` and *Root*`.sim`. It is convenient to describe them here and simply reference them in subsequent sections.

## 3.1 Component Weights

The following records can be put at level 2 for a hull, lifting appendage, or lump. Under some circumstances, they are also valid for an out-of-plane calculation specification.

| `Add` |
|---|

The component loads are given a weight of 1.0, which means they are added to the totals. Since that is currently the default, this record is redundant.

| `Subtract` |
|---|

The component loads are given a weight of –1.0, which means they are subtracted from the totals. See the discussion of component weights in reference [2].

| `Zero` |
|---|

The component loads are given a weight of zero, which means they are ignored in the totals. See the discussion of component weights in reference [2].

## 3.2 Label

| `Label` *label string* | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *label string* | component label | | | |

The label string consists of '`#`' or '`?`' followed by up to 24 alphabetic characters; case is ignored. If there is no `Label` record, the default label is `#NAME`*nnn* where `NAME` is determined by the component type and *nnn* is the number of the component as encountered in the input file, i.e., *nnn* = 001 for the first, 002 for the second, etc. The default name of each component type is indicated after the appropriate input summary table in the following sections.

## 3.3 Pop

| Pop |
|---|

A Pop record forces termination of level 2 or higher input, completing processing of the current set of calculations and returning to the next level down in the input stream (an instance in which the order of input records is important). Normally, an unrecognized keyword (or the end of file) triggers this sequence, but Pop can be used to (a) clearly indicate where this happens and (b) provide a work-around in case there is ambiguity in the sequence of records. (The latter should have been avoided in the current build, but it may get inadvertently introduced in later versions.)

## 3.4 Simulation Input Records

The following records can be input for most of the calculations and simulations on *Root*.sim. DeltaLim and Reference redefine parameters that then remain unchanged until redefined in a subsequent simulation. WeightDCI affects only the current simulation; the weights are reset to their initial values on completion. A Text record is simply output to *Root*.spr when it is encountered.

Note that a number of the concepts referred to here are developed in later sections.

| DeltaLim | *label* or $N_L$, $\delta_{Max}$, $\delta_{Min}$ | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *label* | lifting appendage label | | | |
| $N_L$ | lifting appendage index | – | | |
| $\delta_{Max}$ | maximum control deflection | deg | | 35 |
| $\delta_{Min}$ | minimum control deflection | deg | $\ldots < \delta_{Max}$ | $-35$ |

The equivalent redefinition of lump control deflection limits is described in section 7.3.

| Reference | [*keyword*] $\begin{bmatrix} x_o, & y_o, & z_o \\ [\phi_o, & \theta_o, & \psi_o] \end{bmatrix}$ | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *keyword* | CB or CG | | | |
| $x_o$, $y_o$, $z_o$ | reference origin in vehicle coordinates | m | | |
| $\phi_o$, $\theta_o$, $\psi_o$ | reference axis rotations | deg | | 0, 0, 0 |

The syntax is the same as used in section 4.4.

Despite the recommendation in section 2.2 against a drastic redefinition of the reference, there are no serious consequences from doing so in a **Static** or **Captive** simulation. However, there may be numerical difficulty in subsequent **Free** simulations since they inherit the previous reference by default. **Coefficnt Derivs** calculations will be incorrect if the reference is moved to a location without port-starboard symmetry.

Angular motions and calculated moments are with respect to the redefined reference.

| **Rho** $\rho$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\rho$ | water density | $kg/m^3$ | $990 \leq \ldots \leq 1035$ | note |

This record redefines water density. It is not valid for a **Coefficnt** simulation. The default value of $\rho$ is 1000 (fresh water) for **Static** and **Captive** simulations; it is 1025.2 (standard sea water) for **Free** and **MLD** simulations.

| **Text** *string* | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *string* | text string of up to 64 characters | | | |

**Text** records provide narrative information for the simulation. Each one is output to *Root*.**spr** as it is encountered.

The string may be preceded by one or more delimiters. It is not parsed, and may therefore contain any character, including delimiters and EOR characters.

| **WeightDCI** *label* or $N_L$, $W_{ID}$, $W_{IH}$, [$W_{IR}$] or<br>**WeightDCI** **All**, *keyword* | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *label* | lifting appendage label | | | |
| $N_L$ | lifting appendage index | – | | |
| $W_{ID}$, $W_{IH}$ | weights for indirect depth and heading | – | $-2 \leq \ldots \leq 5$ | 1 |
| $W_{IR}$ | weight for indirect roll | – | $-2 \leq \ldots \leq 5$ | 1 |
| *keyword* | **One**, **Reset**, or **Zero** | | | |

The purpose of this record is discussed in section 6.3. In the form shown here, it is directed only to the indicated appendage, or to all.

**One**: all weights are set to a value of one.

**Reset**: all weights are reset to their initial value.

`Zero`: all weights are set to a value of zero.

Multiple `WeightDCI` records can be used to reset weights for different appendages. The global form of this record, '`WeightDCI All ...`' (only one of which is permitted per simulation), overrides weight values on individual appendages and clears the associated flags. Weights on specific appendages can then be reset individually. For example, to turn on all the appendages to indirect control commands except those numbered two and three, we can input:

```
WeightDCI All One
WeightDCI 2  0 0 0
WeightDCI 3  0 0 0
```

`WeightDCI` records should precede indirect `Delta` commands to which they apply.

Information on the initial status of some of the above quantities, notably the vehicle reference and indirect control weights, is output in the 'Configuration Summary' section at the beginning of file *Root*`.spr`. Most changes that result from resetting these and other parameters are recorded subsequently on the file as they occur.

## 4 Root.geo Level 1 Input

*Root*`.geo` input defines component geometry in simple terms, e.g., hull breadth and height. Since vehicle components are input independently, interference calculations need additional parameters such as the hull radius at the attachment point of an appendage. The locations and characteristics of propulsors and other systems are needed to complete the description. In the first phase of processing, hydrodynamic quantities are calculated for fresh water density, $\rho = 1000$ kg/m$^3$; later, they are scaled for a given density.

*Root*`.geo` level 1 input is used to introduce the vehicle components, including lumps — descriptions of which are then input at Level 2 — and to provide overall program control. Return to Level 1 from Level 2 occurs automatically when an out-of-context keyword is encountered.

The following table summarizes the keyword records at level 1.

| Keyword | Function |
|---|---|
| `AddedMass` | define method for added mass and moment calculations |
| `AutoDepth` | depth autopilot parameter input to follow |
| `AutoHead` | heading autopilot parameter input to follow |
| `AutoProp` | propulsion autopilot parameter input to follow |
| `AutoRoll` | roll autopilot parameter input to follow |

| Keyword | Function |
|---|---|
| `BG` | define **BG** (vector) |
| `CB` | define location of CB |
| `CG` | define location of CG |
| `Comp` | compensation tank data to follow |
| `CompRate` | define compensation flow rate |
| `Compress` | hull and tile compression data to follow |
| `Help` | send help output to *Root*`.log` |
| `HPAir` | HP air bottle group input to follow |
| `Hull` | hull component input to follow |
| `Inertial` | define method for calculation of inertial properties |
| `Lift` | lifting component input to follow |
| `Lump` | lump component input to follow (inactive) |
| `MBT` | main ballast tank (MBT) component input to follow |
| `OOPCalc` | out-of-plane load calculation input to follow |
| `Output` | define output control parameter |
| `Plot` | generate a geometry plot file |
| `Prop` | propulsor component input to follow |
| `Propulsor` | propulsor component input to follow |
| `Reference` | define reference origin and axes |
| `Text` | title or other information |
| `Trim` | trim tank data to follow |
| `TrimComp` | trim/compensation tank data to follow |
| `TrimRate` | define trim flow rate |
| `WagBInit` | initialize Wageningen B propellers |
| `Warnings` | define warning control parameter |
| `WTC` | WTC component input to follow |

`Reference` is required in the absence of a hull; otherwise these records are optional and may in general appear in any order on the file. Note, however that the program assigns indices to lifting and other components in the order that they are encountered, and these indices may be used in later input to identify the components. In addition, the association between BGs and MBTs may be implied by the order of `HPAir` and `MBT` records (section 13).

Here and elsewhere in this reference, record descriptions are grouped to illustrate commonalities and interactions between them.

## 4.1 Added Mass and Inertial Properties

These properties are determined by `AddedMass`, `BG`, `CB`, `CG`, and `Inertial` records. Note that only two of the `BG`, `CB`, and `CG` records are needed to define them all. Providing one or none causes the program to resort to defaults, while providing all three will result in a fatal error if they are inconsistent.

| AddedMass [*keyword*] | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *keyword* | Esam, None, or Simple | | | Esam |

Added masses and added moments of inertia are estimated for hull and lifting components using the method indicated in this record. For lumps they are input as nondimensional coefficients.

**Esam**: components of the **ESAM** code [4] are used in the calculation, and portions of the internal **ESAM** dialogue are sent to *Root*.**log**. In the absence of an **AddedMass** record, this is the default method.

**None**: the calculation is omitted; this option is only valid for **Static** simulations.

**Simple**: use classic methods for equivalent ellipsoids and flat plates.

| BG $[x_{BG},\ y_{BG}],\ \overline{BG}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $x_{BG},\ y_{BG}$ | axial and lateral components of **BG** | m | | 0 |
| $\overline{BG}$ | metacentric height | m | $0 \leq \ldots$ | note |

The full vector form of the metacentric distance is expressed as $\mathbf{BG} = (\mathbf{CG} - \mathbf{CB})$. Normally only its vertical component $z_{BG} = (z_G - z_B) = \overline{BG}$ is required. See the note below on combining **BG**, **CB**, and **CG** records.

A single argument is taken to be $\overline{BG}$, otherwise all three arguments must be present. If there is only one hull component, the $\overline{BG}$ default is $D_M/20$, where $D_M$ is maximum diameter of the hull.

| CB $x_B,\ y_B,\ z_B$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $x_B,\ y_B,\ z_B$ | CB in vehicle coordinates | m | | note |

In the absence of this record, the default center of buoyancy vector **CB** is derived from form displacement.

| CG $x_G,\ y_G,\ z_G$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $x_G,\ y_G,\ z_G$ | CG in vehicle coordinates | m | | note |

The three components of CG are given in vehicle axes. In the absence of this record, the default center of gravity vector **CG** is derived from the **BG** and **CB** defaults.

| `Inertial` | [*keyword*] | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *keyword* | `Form` or `None` | | | `Form` |

`Form`: form displacement (or volume and location for lumps), assuming neutral buoyancy, is used to estimate the inertial properties. In the absence of an `Inertial` record, this is the default method.

`None`: the calculation is omitted; this option is only valid for `Static` simulations.

## 4.2 Component Specification

Up to four hull components, twelve lifting appendage components, and four lump components can be specified. They are announced in input by `Hull`, `Lift`, and `Lump` records respectively. The out-of-plane specification, announced by `OOPCalc`, is included here because it is intimately related to hulls and appendages. Each of these records is followed by multiple level 2 input records described in subsequent sections.

| `Hull` | [*keyword*, [$N_S$]] | | | → |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *keyword* | `Default`, `HulFoM`, `HulFor`, or `Jeans` | | | `HulFoM` |
| $N_S$ | number of hull stations | – | $11 \leq \ldots \leq 51$ | 21 |

This record introduces a new hull component. Level 2 hull input records are described in section 5.

`Default`: a placemarker for future developments in hull load calculations. At present it is equivalent to `HulFoM`.

`HulFoM`: the empirical method of reference [5] is used to calculate translational loads and the `HulFor` method [6] is used to calculate rotational loads; this is currently the default.

`HulFor`: the `HulFor` method [6] is used to calculate both translational and rotational loads. This option is deprecated.

`Jeans`: the Jeans estimation method [7] is used to calculate translational loads and the `HulFor` method [6] is used to calculate rotational loads.

The second optional argument, $N_S$, must be preceded by the first, which is itself optional. However, the default value for $N_S$ appears to be satisfactory for typical submarine hulls.

| `Lift` | → |
|---|---|

This record introduces a new lifting appendage. The appropriate level 2 input records are described in section 6.

| `Lump` → |
|---|

This record introduces a new lump component. The appropriate level 2 input records are described in section 7.

| `OOPCalc None` or | | | | |
|---|---|---|---|---|
| `OOPCalc` | | | | → |
| Argument(s) | What | Units | Range | Default |
| `None` | suppress out-of-plane calculations | | | |

This record allows the user to manipulate out-of-plane load calculations for hulls and lifting components. If it is absent, a calculation is done for each lifting appendage with the type `Sail`, using the nearest hull in each case.

Without the argument `None`, only calculations specified in subsequent level 2 input are done; this input is described in section 8.

`None`: with this argument, all out-of-plane calculations are suppressed.

### 4.3 Propulsion

Up to three propulsors can be defined. Each requires a `Prop` or `Propulsor` record (the two are treated identically). Propulsor type specifies the method of estimating propulsor thrust and torque coefficients. A `WagBInit` record and associated level 2 input enable the user to override defaults used for initial optimization of `WageningB`-type propulsors.

| `Prop` [*keyword*] or | | | | → |
|---|---|---|---|---|
| `Propulsor` [*keyword*] | | | | → |
| Argument(s) | What | Units | Range | Default |
| *keyword* | `DSSPtwo`, `STRprop`, or `WageningB` | | | `WageningB` |

Propulsor type is defined by the optional argument. The default is `WageningB`.

`DSSPtwo`: a 'small main propulsor' propeller model approximating that used for the Oberon class in `DSSP02` [8]; i.e., for an older twin-screw submarine arrangement. It may not represent single screw propulsion particularly well since the propeller inflow conditions are different.

`STRprop`: a 'large main propulsor' propeller model based on the four-quadrant single screw behind-model wind tunnel data reported in reference [9]. It has been observed

that the propulsive efficiency of this model is unrealistically high at large advance ratio. This option is deprecated.

**WageningB**: a 'large main propulsor' propeller model based on two-quadrant open water data for the Wageningen B4–70 series [10]. The method includes semiempirical calculations for wake fraction and thrust deduction [11], and approximates optimum pitch/diameter ratio.

Note that mixing propulsor types is not recommended; in any case, **WageningB** propulsors cannot be mixed with other types. Level 2 propulsor input records are described in section 9.

| **WagBInit** → |
|---|

This record introduces data for initializing **WageningB** propulsors. The appropriate level 2 input records are described in section 10.

## 4.4 The Hydrodynamic Reference

A **Reference** record is required to set (or override the default, see section 3.4) reference axes and origin in *Root*.**geo**. Furthermore, a new reference definition, which will supercede this one, can be supplied with simulation data in *Root*.**sim**. At any stage, it will avoid problems if the reference origin is in the vicinity of the vehicle CB and CG, and if the reference axes correspond to conventional hydrodynamic axes.

**Reference** [*keyword*]

$$\begin{bmatrix} x_o, & y_o, & z_o \\ [\phi_o, & \theta_o, & \psi_o] \end{bmatrix}$$

| Argument(s) | What | Units | Range | Default |
|---|---|---|---|---|
| *keyword* | CB or CG | | | |
| $x_o$, $y_o$, $z_o$ | reference origin in vehicle coordinates | m | | |
| $\phi_o$, $\theta_o$, $\psi_o$ | reference axis rotations | deg | | 0, 0, 0 |

If this record has the keyword CB or CG as an argument then there is no other input and the reference origin is located at the CB or CG of the vehicle with the reference axes directed along the vehicle axes.

Otherwise, the reference origin and, optionally, reference axes (angles of rotation from vehicle axes, in reverse order) must follow as triplets of real numbers in a two-record data block.

## 4.5 Submarine Systems

This rather loosely defined category comprises autopilots, compression effects, high pressure (HP) air, main ballst tanks (MTBs), watertight compartments (WTCs), and the trim and compensation systems. Many of these are indexed and labeled like hull and other components. There can be up to nine HP air bottle groups, eight MBTs, eight WTCs, and twelve trim or compensation tanks.

| | | | |
|---|---|---|---|
| `AutoDepth` | [*keyword*] | or | → |
| `AutoHead` | [*keyword*] | or | → |
| `AutoProp` | [*keyword*] | or | → |
| `AutoRoll` | [*keyword*] | | → |

| Argument(s) | What | Units | Range | Default |
|---|---|---|---|---|
| *keyword* | `PhaseLead` or `PID` | | | note |

There are autopilots available to control depth, heading, propulsion, and roll. They simplify free manoeuvers, e.g., by automatically maintaining depth during a turn, and may also be used in MLD recoveries. These records introduce parameter input for the corresponding autopilot. The optional keyword argument indicates a default autopilot type. In its absence the depth, heading, and roll autopilots default to `PhaseLead`, and the propulsion autopilot to `PID`. Optional level 2 input records that follow are described in section 11. Use of the autopilots in a free simulation is described in section 26.2.

`PhaseLead`: the autopilot uses a phase lead algorithm to relate the error and control signals.

`PID`: the autopilot uses a PID (Proportional-Integral-Derivative) algorithm to relate the error and control signals.

| | |
|---|---|
| `Comp` | → |

This record introduces a new compensation tank. The level 2 `Comp` input records are described in section 15.

| `CompRate` | $\dot{V}_{ct}$ | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\dot{V}_{ct}$ | compensation pump volume flow rate | $m^3/s$ | $0 \leq \ldots$ | |

There is one compensation pump available to transfer water between `Comp` or `TrimComp` tanks and the sea. Its flow rate is independent of depth.

| `Compress` → |
|---|

This record introduces compression data for the hull, resilient tiles, or both. The level 2 `Compress` input records are described in section 12.

| `HPAir` → |
|---|

This record introduces a new HP air bottle group. The level 2 `HPAir` input records are described in section 13.1.

| `MBT` → |
|---|

This record introduces a new main ballast tank. The level 2 `MBT` input records are described in section 13.2.

| `Trim` → |
|---|

This record introduces a new trim tank. The level 2 `Trim` input records are described in section 15.

| `TrimComp` → |
|---|

This record introduces a new trim/compensation tank. The level 2 `TrimComp` input records are described in section 15.

| `TrimRate` $\dot{V}_{tt}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\dot{V}_{tt}$ | trim pump volume flow rate | $m^3/s$ | $0 \leq \ldots$ | |

There is one trim pump available to transfer water between `Trim` or `TrimComp` tanks. Its flow rate is independent of depth.

| `WTC` → |
|---|

This record introduces a new watertight compartment. The level 2 `WTC` input records are described in section 14.

## 4.6 Other Input

| `Help` |
|---|

When a `Help` record is encountered, some help text is sent to *Root*`.log`, and the program is stopped. The help text consists principally of keyword dictionary listings for the current program build.

| `Output` *keyword* | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *keyword* | `Normal`, `Quiet`, or `Verbose` | | | `Normal` |

This record is to control the amount of information sent to *Root*`.log`. In *Root*`.geo`, it applies only to the geometry and systems setup phase of execution. Since `Output` only affects output that is generated after it is read, it should be at or near the start of the file.

`Normal`: is the default for output unless there is a different specification in `DSSP50.myp`, see section 30.

`Quiet`: suppresses some extraneous output. In the present build it does not make a lot of difference.

`Verbose`: should not be necessary unless a problem is encountered. It adds a lot of detail that may be useful for troubleshooting, but makes *Root*`.log` hard to read. In one or two cases `Verbose` activates additional warnings.

| `Plot` [$r_{view}$] | | | | | → |
|---|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default | |
| $r_{view}$ | initial viewing distance for VRML rendering | m | | note | |

This record introduces level 2 geometry plotting information as described in section 13. Without it the plot file *Root*`.gpl` is not created.

The optional argument $r_{view}$ determines the viewing distance for initially rendering VRML images in some viewing software; in the present build it is ignored for other plotting formats. By default $r_{view}$ is the length of the hull, if there is only one; otherwise it is zero.

| `Text` *string* | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *string* | text string of up to 64 characters | | | |

**Text** records provide title and other information for the simulation. Each one is output to *Root*.**gpr** as it is encountered, and the first ten **Text** records are saved as heading information for the simulation phase of program execution.

The string may be preceded by one or more delimiters. It is not parsed, and may therefore contain any character, including delimiters and EOR characters.

| **Warnings** | *keyword* | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *keyword* | **Anal**, **Normal**, or **Quiet** | | | **Normal** |

This record is used to control how a warning with pause [1] is handled. In *Root*.**geo**, it applies only to the geometry and systems setup phase of execution. **Warnings** only affects warning messages that are generated after it is read, so it should be at or near the start of *Root*.**geo**. Once any issues with *Root*.**geo** input have been resolved, it will be preferable to run the program with **Warnings** set to **Quiet**.

**Anal**: any warning with pause is upgraded to a fatal error.

**Normal**: is the default for handling a warning with pause unless there is a different specification in **DSSP50.myp**, see section 30.

**Quiet**: any warning with pause is downgraded to a regular warning.

## 5 Root.geo Level 2 Hull Input

The nomenclature used to define basic hull component geometry is illustrated in figure 3. A hull is defined at a series of stations between the nose and tail. These data may be interpolated to a different set of stations. The minimum information required is diameter at each station for an axisymmetric hull, and a direct or indirect definition of the nose and tail points in vehicle axes. The tail section may be closed, with zero diameter, or truncated, with a nonzero diameter. In the latter case, base drag can be modeled. Additional optional input for each station includes breadth and height (in the case of non-axisymmetry), cross section area, and camber (which is only used for plotting geometry in this version of the program).

Values of residual drag coefficient ($C_{\Delta f}$) or roughness allowance ($k_r$) may be input to override the default values. Lateral loads are increased if there is a deck casing or keel. There is no reliable way to estimate these increases at present; the user must supply appropriate factors on the **Deck** and **Keel** records.

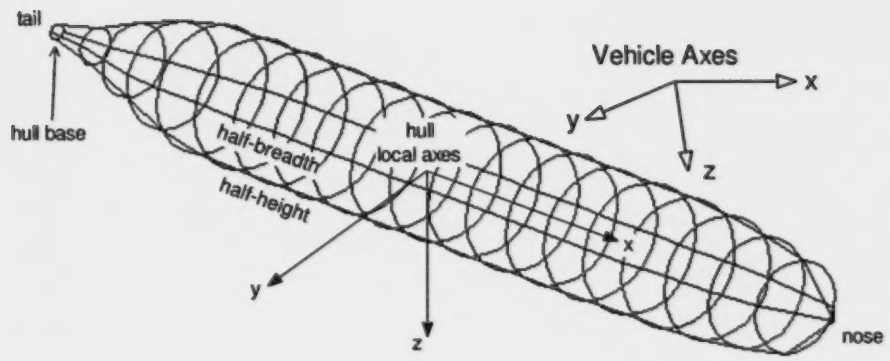


**Figure 3.** Hull definition nomenclature.

The following table summarizes the keyword records used to characterize hull components.

| Keyword | Function |
|---|---|
| `Add` | component loads are added to the total, see section 3.1 |
| `BaseArea` | define hull base area |
| `Deck` | adjust sideforce and yawing moment for a deck casing |
| `Diameter` | diameter data block to follow |
| `HullSep` | force afterbody separation |
| `Keel` | adjust sideforce and yawing moment for a keel |
| `Label` | component label, see section 3.2 |
| `Length` | define component length |
| `LoD` | define length/diameter ratio |
| `MidHull` | define midpoint |
| `Nose` | define nose point |
| `Pitch` | define rotation in pitch |
| `Pop` | force termination of input for this component, see section 3.3 |
| `ResiDragR` | define hull residual drag coefficient or method of calculation |
| `Roll` | define rotation in roll |
| `RoughAllow` | define hull roughness allowance |
| `Source` | identify the data source for `HulFoM` load estimates |
| `Station` | station data block to follow |

| | |
|---|---|
| **Subtract** | component loads are subtracted from the total, see section 3.1 |
| **Tail** | define tail point |
| **Yaw** | define rotation in yaw |
| **Zero** | component loads are ignored in the total, see section 3.1 |

The default label for a hull is **#HULL***nnn* where *nnn* is the number of the hull as encountered on the input file.

A keyword record other than one of these will terminate input for the current component. After processing level 2 data is completed, the input stream is backspaced and the new record is re-read at level 1. The records are discussed below in terms of the functional group that they belong to.

## 5.1 Hull Geometry

Hull geometry, including its location and attitude in the vehicle coordinate system is specified with a combination of the following records: **Diameter**, **Length**, **MidHull**, **Nose**, **Pitch**, **Roll**, **Station**, **Tail**, and **Yaw**.

The local $x$ axis of the hull is directed forward and passes through the nose point, midpoint, and tail point. (The mid point is simply the mean of the nose and tail points.) To fully define hull location and attitude, the user must provide an unambiguous specification; redundant data are checked for consistency, and inconsistency generates a fatal error. The axis rotations are done in order $\Psi$, $\Theta$, $\Phi$.

**Diameter** $[N_{S_{in}}]$
$B_1$, $B_2$, $B_3$
$B_4$, $B_5$, etc.

| Argument(s) | What | Units | Range | Default |
|---|---|---|---|---|
| $N_{S_{in}}$ | number of values in data block following | – | $0 < \ldots \leq 51$ | 21 |
| $B_i$ | diameter (breadth) at i$^{th}$ station | m | $0 \leq \ldots$ | |

The hull is axisymmetric with diameter, or breadth, specified at $N_{S_{in}}$ equally spaced stations from nose to tail. The data block must have three entries per record, apart from the last, which may have fewer.

In generating the hull geometry, $T_i = B_i$ and $A_i = \pi B_i^2/4$, where $T$ is height and $A$ is cross section area. Non-axisymmetric hull sections ($T_i \neq B_i$), or those for unequal spacing, should be specified with a **Station** record, below.

If the number of stations for calculation, $N_S$, optionally specified on the **Hull** record, is not equal to $N_{S_{in}}$, the **Diameter** data will be interpolated. (The **Hull** record must have been given both its arguments in order to specify $N_S$, e.g., '**Hull Default** $N_S$'.)

| Length $L$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $L$ | hull length | m | $0 < \ldots$ | |

**Length** plus one of **Nose**, **MidHull**, or **Tail** provides an unambiguous hull axis definition for the default (zero) rotation. Rotation angles may be defined as required to override the default values.

| MidHull $x_m$ , $[y_m$ , $z_m]$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $x_m$ | axial coordinate of mid hull | m | | |
| $y_m$, $z_m$ | lateral and normal coordinates of mid hull | m | | 0, 0 |

The coordinates are with respect to vehicle axes. **MidHull** plus one of **Nose** or **Tail** provides an unambiguous definition of the hull axis including yaw and pitch rotation angles; a **Roll** record is explicitly required to override the roll angle default. **MidHull** plus **Length** assumes the default (zero) rotation angles unless one or more of **Yaw**, **Pitch**, or **Roll** is also provided.

| Nose $x_n$ , $[y_n$ , $z_n]$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $x_n$ | axial coordinate of nose | m | | |
| $y_n$, $z_n$ | lateral and normal coordinates of nose | m | | 0, 0 |

The coordinates are with respect to vehicle axes. **Nose** plus one of **MidHull** or **Tail** provides an unambiguous definition of the hull axis including yaw and pitch rotation angles; a **Roll** record is explicitly required to override the roll angle default. **Nose** plus **Length** assumes the default (zero) rotation angles unless one or more of **Yaw**, **Pitch**, or **Roll** is also provided.

| Pitch $\Theta$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\Theta$ | pitch with respect to the vehicle axis system | deg | | 0 |

Rotations are taken in conventional order: yaw, pitch, roll. A value for $\Theta$ will be calculated if at least two records with the keywords **Nose**, **MidHull**, or **Tail** are present.

| **Roll** $\Phi$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\Phi$ | roll with respect to the vehicle axis system | deg | | 0 |

Rotations are taken in the conventional order: yaw, pitch, roll.

**Station** $[N_{S_{in}}]$
$B_1$, $[T_1$, $[A_1$, $[F_1]]]$
$B_2$, $[T_2$, $[A_2$ $[F_2]]]$
etc. or

**Station** $N_{S_{in}}$ , **Irregular**
$x_{in_1}$, $B_1$, $[T_1$, $[A_1$, $[F_1]]]$
$x_{in_2}$, $B_2$, $[T_2$, $[A_2$ $[F_2]]]$
etc.

| Argument(s) | What | Units | Range | Default |
|---|---|---|---|---|
| $N_{S_{in}}$ | number of records in data block following | – | $0 < \ldots \leq 51$ | 21 |
| $x_{in_i}$ | axial coordinate of i$^{th}$ station | – | | |
| $B_i$ | breadth at i$^{th}$ station | m | $0 \leq \ldots$ | |
| $T_i$ | height at i$^{th}$ station | m | $0 \leq \ldots$ | note |
| $A_i$ | area at i$^{th}$ station | m$^2$ | $0 \leq \ldots$ | note |
| $F_i$ | camber at i$^{th}$ station | m | | note |

Note that axisymmetric hulls with equally spaced stations are input more simply using the **Diameter** record, above. If the station spacing is equal, $x_{in_i}$ is not required and $N_{S_{in}}$ is optional. Otherwise, $N_{S_{in}}$, the argument **Irregular**, and $x_{in_i}$ in the data block are all required.

Stations are ordered from nose to tail. To simplify input the axial coordinate is automatically scaled to hull length, so that input may run from $x_{in_i} = 0$ (nose) to 1 (tail), for example.

Hull camber, by analogy with appendage section camber, is station mid-height offset from the nose–tail line, positive downwards. If $F$ is omitted, $F_i = 0$. If $A$ is omitted, $A_i = \pi B_i T_i/4$, and if $T$ is omitted, $T_i = B_i$ and $A_i = \pi B_i^2/4$.

If the number of stations for calculation, $N_S$, optionally specified on the **Hull** record, is not equal to $N_{S_{in}}$, the data will be interpolated. (The **Hull** record must have been given both its arguments in order to specify $N_S$, e.g., '**Hull Default** $N_S$'.)

| Tail $x_t$ , $[y_t$ , $z_t]$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $x_t$ | axial coordinate of tail | m | | |
| $y_t$, $z_t$ | lateral and normal coordinates of tail | m | | 0, 0 |

The coordinates are with respect to vehicle axes. `Tail` plus one of `Nose` or `MidHull` provides an unambiguous definition of the hull axis including yaw and pitch rotation angles; a `Roll` record is explicitly required to override the roll angle default. `Tail` plus `Length` assumes the default (zero) rotation angles unless one or more of `Yaw`, `Pitch`, or `Roll` is also provided.

| Yaw $\Psi$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\Psi$ | yaw with respect to the vehicle axis system | deg | | 0 |

Rotations are taken in the conventional order: yaw, pitch, roll. A value for $\Psi$ will be calculated if at least two records with the keywords `Nose`, `MidHull`, or `Tail` are present.

## 5.2 Additional Parameters

Records `BaseArea`, `Deck`, `HullSep`, `Keel`, `LoD`, `ResiDragR`, `RoughAllow`, and `Source` define additional hull parameters.

| BaseArea $A_B$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $A_B$ | hull base area | $m^2$ | $0 \leq \ldots \leq \max A_i$ | note |

To redefine $A_B$ for hull base drag calculations. Otherwise, $A_B = A_{N_S}$, the base area input in `Diameter` or `Station`.

| Deck $\eta_{Yd}$, $\eta_{Nd}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\eta_{Yd}$ | factor for sideforce | – | $0 \leq \ldots$ | 1 |
| $\eta_{Nd}$ | factor for yawing moment | – | $0 \leq \ldots$ | 1 |

Load corrections for the presence of a deck casing. The hydrodynamic sideforce will be multiplied by $\eta_{Yd}$ and the yawing moment by $\eta_{Nd}$. While there are no systematic data from which to select these corrections, reference [5] illustrates a case for which values

of (1.2, 1.0) may be appropriate. The user may be able to deduce suitable values by tuning predictions with trials data.

| `HullSep` $N_{sep}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $N_{sep}$ | hull station for flow separation to occur | – | | |

Forces hull separation to occur at hull station $N_{sep}$. The value is automatically clipped to be within the first to last stations, 1 to $N_S$.

Specifying $N_{sep}$ allows the user to address a '`No minimum found in dA/dx (DT2) for separation`' error [1], or to override the estimated location of separation, which is output on the *Root*`.log` file.

| `Keel` $\eta_{Yk}$, $\eta_{Nk}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\eta_{Yk}$ | factor for sideforce | – | $0 \leq \ldots$ | 1 |
| $\eta_{Nk}$ | factor for yawing moment | – | $0 \leq \ldots$ | 1 |

Load corrections for the presence of a keel. The hydrodynamic sideforce will be multiplied by $\eta_{Yk}$ and the yawing moment by $\eta_{Nk}$. While there are no systematic data from which to select these corrections, reference [5] illustrates a case for which values of (1.7, 0.9) may be appropriate. The user may be able to deduce suitable values by tuning predictions with trials data.

| `LoD` $L/D_M$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $L/D_M$ | hull length/diameter ratio | – | $4 \leq \ldots \leq 16$ | |

This record overrides the calculated value of length/diameter ratio for the current hull component.

| `ResiDragR` $C_{\Delta f}$ or<br>`ResiDragR` *keyword* | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $C_{\Delta f}$ | hull residual drag coefficient | – | $0 \leq \ldots \leq 0.5$ | note |
| *keyword* | `Hoerner` or `Torpedo` | | | |

In the absence of this record, a semiempirical expression given by Hoerner [12] for bodies of rotation is used to calculate $C_{\Delta f}$.

`Hoerner`: forces the Hoerner calculation, and is presently redundant.

`Torpedo`: forces a fit to experimental data for torpedo-like bodies.

| `RoughAllow` $k_r$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $k_r$ | hull roughness allowance | – | $0 \leq \ldots \leq 0.001$ | 0.0004 |

This record overrides the default hull roughness allowance in the hull drag calculation.

| `Source` *keyword* | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *keyword* | `Kriging` or `NeuralNet` | | | `Kriging` |

The `HulFoM` (default) hull translational load calculations use lookup tables derived from model test data. There are two sets of tables in the present build, one derived from Kriging interpolation, the other from neural network generalization [5].

`Kriging`: forces use of the Kriging interpolation tables, and is presently redundant.

`NeuralNet`: forces use of the neural network tables.

## 6 Root.geo Level 2 Lift Input

The nomenclature used to define basic lifting appendage geometry is illustrated in figure 4. The four corner points $c1, c2, c3$, and $c4$ must be given in vehicle axes. Appendage sections are assumed to be NACA 4–digit, and thickness/chord ratio and other parameters are optional. An isolated appendage is assumed to be mounted on a reflecting plane at the root, i.e., loading is finite at the root because of lift carry-over. An unattached symmetrical wing would therefore be modeled by two isolated appendages, one for each semispan. All other appendage configurations require a type specification (`Rudder`, `Sail`, etc.) in order to model interference effects.

Figures 5, 6, and 7 illustrate a number of the additional geometric parameters needed to account for interference. In addition, number of non-geometric parameters may be defined where it is necessary to override default values assigned by the program.

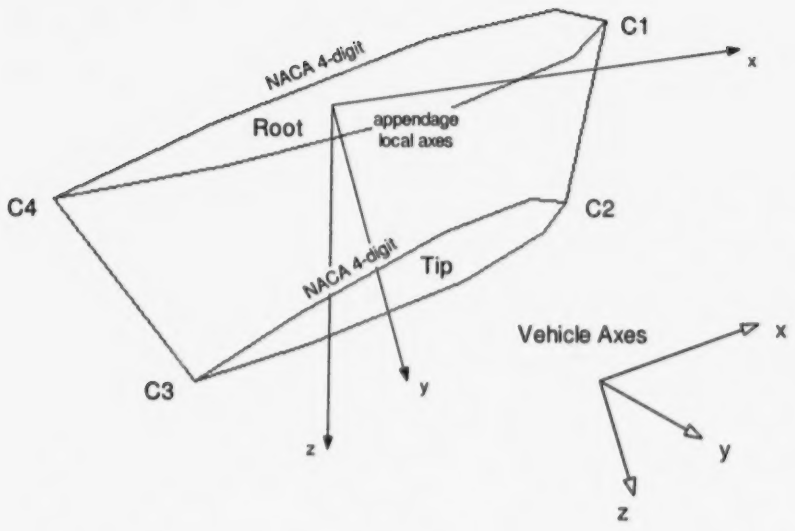


**Figure 4.** Appendage definition nomenclature.

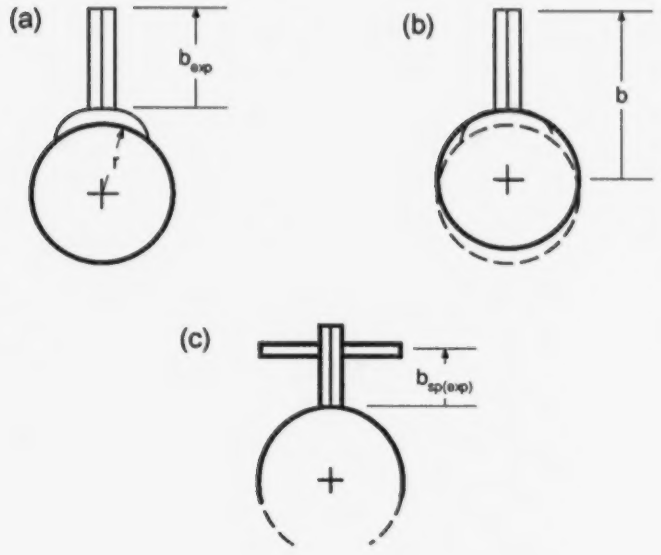


**Figure 5.** Hull/sail input nomenclature: (a) non-axisymmetric hull, (b) equivalent axisymmetric hull, and (c) sailplane location.

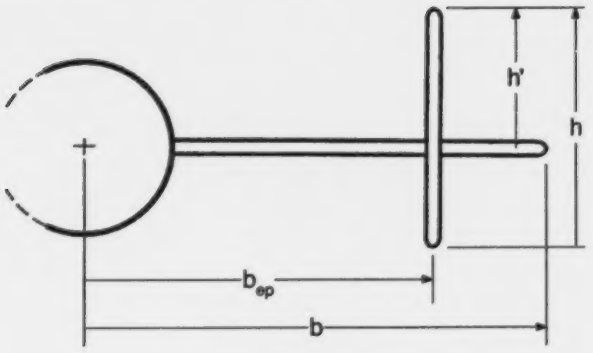


**Figure 6.** Endplate input nomenclature.

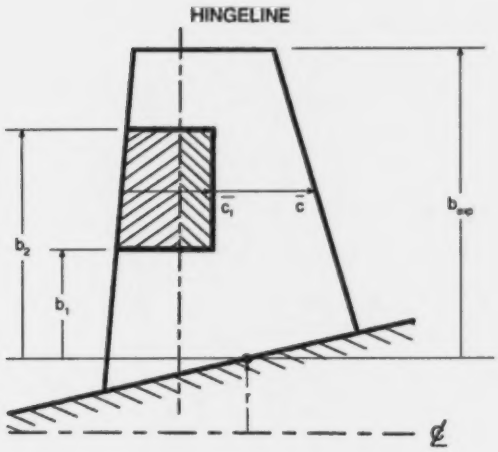


**Figure 7.** Flapped appendage input nomenclature.

The following table summarizes the keyword records used to characterize lifting appendages:

| Keyword | Function |
|---|---|
| `Add` | component loads are added to the total, see section 3.1 |
| `AllMoving` | define control type as All-Moving |
| `Camber` | apply geometric camber |
| `CXJ` | define junction axial force coefficient |
| `CXZero` | define zero lift axial force coefficient |
| `DelNonLin` | enable nonlinear flap calculation |
| `DelRateLim` | set control deflection rate limit |
| `Del` | set initial control deflection |
| `Delta` | set initial control deflection |
| `DeltaDyn` | set control dynamic parameters |
| `DeltaLim` | set control deflection limits |
| `FlapGap` | define flap balance and gap correction |
| `Flapped` | define control type as Flapped |
| `FlaProDrag` | define flap profile drag |
| `HasEndP` | correct for having an endplate on this appendage |
| `IsEndP` | correct for this appendage being an endplate |
| `Kill` | save for a copy, but do not use, this appendage |
| `Label` | component label, see section 3.2 |
| `LBowplane` | define appendage type as Large Bowplane |
| `Planform` | appendage corner data block follows |
| `Pop` | force termination of input, see section 3.3 |
| `PReverse` | plane reversal data follows |
| `Reflected` | copy saved appendage by reflecting |
| `Rotated` | copy saved appendage by rotating |
| `Rudder` | define appendage type as Rudder |
| `Sail` | define appendage type as Sail |
| `Sailplane` | define appendage type as Sailplane |
| `Save` | save this appendage for subsequent copy |
| `SBowplane` | define appendage type as Small Bowplane |
| `ShipStab` | define appendage type as Ship Stabilizer |
| `Sternplane` | define appendage type as Sternplane |
| `Subtract` | component loads are subtracted from the total, see section 3.1 |
| `Tail` | define appendage type as Tail |
| `TailEff` | define tail efficiency |
| `ToC` | define section thickness/chord ratio |
| `Twist` | apply geometric twist |
| `WeightDCI` | set indirect control deflection weights |
| `Zero` | component loads are ignored in the total, see section 3.1 |

The default label for an appendage is #LIFT*nnn* where *nnn* is the number of the appendage as encountered on the input file.

A keyword record other than one of these will terminate input for the current component. Pop is described in section 4.2. After processing of level 2 data is completed, the input stream is backspaced and the new record is re-read at level 1. The records are described below in terms of the functional group that they belong to.

## 6.1 Appendage Geometry

The appendage planform is described by four corner points as illustrated in figure 4. For a triangular appendage, points *c*2 and *c*3 should be made coincident. The user can additionally override default values of camber, thickness/chord ratio, and twist, and specify whether the appendage has, or is, an endplate.

| **Camber** $f$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $f$ | nondimensional camber | – | | 0 |

Section camber $f$ is positive in the sense of local $z$. Here it is nondimensionalized by local chord; the ratio is assumed constant across the span of the appendage.

| **HasEndP** $h/b$, $[b_{ep}/b]$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $h/b$ | height/semispan ratio of plate | – | $0 \leq \ldots \leq 2.5$ | |
| $b_{ep}/b$ | distance/semispan ratio of plate | – | $0 \leq \ldots \leq 1$ | 1 |

The current appendage has an endplate. Parameters $h$, $b$, and $b_{ep}$ are endplate height, appendage semispan, and spanwise location of the endplate (the last two measured from the hull centerline if appropriate), see figure 6.

| **IsEndP** $[h'/h]$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $h'/h$ | attachment/height ratio of plate | – | $0 \leq \ldots \leq 1$ | 0.5 |

The current appendage is an endplate. Parameters $h'$ and $h$ are point of attachment relative to the full span and the full span ($h = 2b$), or height, see figure 6.

| Planform | | | | |
|---|---|---|---|---|
| $x_{c1}$, $y_{c1}$, $z_{c1}$ | | | | |
| $x_{c2}$, $y_{c2}$, $z_{c2}$ | | | | |
| $x_{c3}$, $y_{c3}$, $z_{c3}$ | | | | |
| $x_{c4}$, $y_{c4}$, $z_{c4}$ | | | | |
| **Argument(s)** | **What** | **Units** | **Range** | **Default** |
| $x_{ci}$, $y_{ci}$, $z_{ci}$ | corner point $ci$ in vehicle coordinates | m | | |

The corner points are input in a data block of four records, one point per record, and must be in order: root leading edge, tip leading edge, tip trailing edge, and root trailing edge. See figure 4.

| ToC $t/c$ | | | | |
|---|---|---|---|---|
| **Argument(s)** | **What** | **Units** | **Range** | **Default** |
| $t/c$ | thickness/chord ratio | – | $0 \leq \ldots \leq 0.25$ | 0.12 |

Section thickness/chord ratio $t/c$ is assumed constant across the span of the appendage. In the absence of this record it is equal to 0.12.

| Twist $\alpha_b$ | | | | |
|---|---|---|---|---|
| **Argument(s)** | **What** | **Units** | **Range** | **Default** |
| $\alpha_b$ | geometric section twist | deg | $-30 \leq \ldots \leq 30$ | 0 |

Geometric section twist is defined as the angle of attack at the tip of the appendage relative to the root; without this record it is equal to zero. Twist is positive in the sense of rotating the appendage local $z$ axis into the $x$ axis.

This record is ignored, with a warning, for triangular planforms and for planforms with implicit twist in the `Planform` specification.

## 6.2 Appendage Type

The keywords which define appendage type are **`LBowplane`**, **`Rudder`**, **`Sail`**, **`Sailplane`**, **`SBowplane`**, **`ShipStab`**, **`Sternplane`**, and **`Tail`**. Defining the appendage type ensures that appropriate interference corrections are made to the appendage loads, and may have additional implications. For example, although interference corrections are identical for types **`Rudder`**, **`Sternplane`**, and **`Tail`**, these types respond differently to control commands. Only one appendage type record may be supplied for each lifting component; if no type is specified, it is treated as an isolated appendage.

These records have at least one argument: a parameter $r$ that is used to estimate lift carry-over. For the general case of an appendage mounted on a hull, and where local dihedral is minimal (i.e., the appendage is more-or-less normal to the hull surface), $r$ is the hull radius at the appendage mid root.

| `LBowplane` $r$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $r$ | mid-root hull radius | m | $0 \leq \ldots$ | |

The appendage is defined as a large bowplane. Whether a bowplane is considered large or small depends on the ratio of its dimensions to the local hull radius [13]. A small bowplane is corrected for dihedral (below), a large one is not.

| `Rudder` $r$, $[r_M]$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $r$ | mid-root hull radius | m | $0 \leq \ldots$ | |
| $r_M$ | maximum hull radius | m | $0 \leq \ldots$ | $r$ |

The appendage is defined as a rudder. Optional argument $r_M$ is required for Dempsey's calculation of tailplane efficiency [14]; otherwise, tailplane efficiency may be specified with a `TailEff` record.

| `Sail` $r$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $r$ | mid-root hull radius | m | $0 \leq \ldots$ | |

The appendage is defined as a sail. Parameter $r$ is usually taken to be the half-breadth of the hull.

| `Sailplane` $r$, $b_{sp(exp)}/b_{exp}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $r$ | mid-root to sail centerline distance | m | $0 \leq \ldots$ | |
| $b_{sp(exp)}/b_{exp}$ | exposed height/span ratio | m | $0 \leq \ldots \leq 1$ | |

The appendage is defined as a sailplane. Parameter $r$ is about half the sail thickness.

Parameters $b_{sp(exp)}$ and $b_{exp}$ are exposed sailplane height (i.e, above the sail root) and exposed sail span, see figure 5.

| SBowplane $r$, $[\phi]$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $r$ | mid-root hull radius | m | $0 \leq \ldots$ | |
| $\phi$ | dihedral/anhedral angle | deg | $-90 \leq \ldots \leq 90$ | 0 |

The appendage is defined as a small bowplane. Optional argument $\phi$ is the dihedral or anhedral angle (sign is immaterial) with respect to the hull normal.

| ShipStab $r$, $[x']$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $r$ | mid-root hull radius | m | $0 \leq \ldots$ | |
| $x'$ | downstream distance | m | $0 \leq \ldots$ | 0 |

The appendage is defined as a ship stabilizer. Optional argument $x'$ is downstream distance between the origin of the hull boundary layer (typically, near the forwardmost point of the hull) and the origin of the appendage axis system.

| Sternplane $r$, $[r_M]$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $r$ | mid-root hull radius | m | $0 \leq \ldots$ | |
| $r_M$ | maximum hull radius | m | $0 \leq \ldots$ | $r$ |

The appendage is defined as a sternplane. Optional argument $r_M$ is required for Dempsey's calculation of tailplane efficiency [14]; otherwise, tailplane efficiency may be specified with a `TailEff` record.

| Tail $r$, $[r_M]$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $r$ | mid-root hull radius | m | $0 \leq \ldots$ | |
| $r_M$ | maximum hull radius | m | $0 \leq \ldots$ | $r$ |

The appendage is defined to be a tail fin. Optional argument $r_M$ is required for Dempsey's calculation of tailplane efficiency [14]; otherwise, tailplane efficiency may be specified with a `TailEff` record.

## 6.3 Control by Appendage Deflections

In order to generate control forces, an appendage is all-moving or has a trailing-edge flap. The following records, which apply to either control type unless otherwise noted,

are used to specify control surface geometry, its dynamics, and other characteristics. See also `TailEff` in the next section.

| `AllMoving` $[S_m/S_{exp}]$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $S_m/S_{exp}$ | all-moving area ratio | – | $0.1 \leq \ldots \leq 1$ | 1 |

The appendage is an all-moving control surface. Parameters $S_m$ and $S_{exp}$ are the area of the moving part and the total exposed area of the appendage.

| `Del` $\delta_\bullet$ or `Delta` $\delta_\bullet$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\delta_\bullet$ | initial control deflection | deg | | 0 |

This record defines an initial, or bias, value for control deflection. The effect of an initial value is to offset the reference deflection for some control commands. The value can be redefined at several places in the *Root*`.sim` file.

| `DelNonLin` |
|---|

This record enables the calculation of effective flap angle, $\delta_e < \delta$, for a flapped appendage [13]. Without the `DelNonLin` record, $\delta_e = \delta$.

| `DelRateLim` $\dot{\delta}_M$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\dot{\delta}_M$ | maximum control deflection rate | deg/s | $0 < \ldots$ | 9 |

Without this record, the maximum deflection rate defaults to 9 deg/s.

| `DeltaDyn` $\zeta_\delta$, $\omega_\delta$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\zeta_\delta$ | control deflection damping | – | $0 < \ldots$ | |
| $\omega_\delta$ | control response frequency | rad/s | $0 < \ldots$ | |

The two arguments, together with $\dot{\delta}_M$, determine control deflection time histories by way of a second order ODE for each control appendage. $\zeta_\delta$ is nondimensional; a warning is issued if it is greater than one.

In the absence of this record, the arguments default to 0.9 and $20\dot{\delta}_M/(\delta_{Max} - \delta_{Min})$ respectively. The latter is equivalent to 10 × maximum rate / |maximum deflection|.

| `DeltaLim` | $\delta_{Max}$, $\delta_{Min}$ | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\delta_{Max}$ | maximum control deflection | deg | | 35 |
| $\delta_{Min}$ | minimum control deflection | deg | $\ldots < \delta_{Max}$ | $-35$ |

These limits can be redefined at several places in the *Root*`.sim` file.

| `FlapGap` | $\eta$ | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\eta$ | flap gap and balance efficiency | – | $0.8 \leq \ldots \leq 1.25$ | 1 |

Flap gap and balance efficiency appears in Lyons and Bisgood's empirical predictions, see reference [13].

| `Flapped` | $[b_1/b_{exp},\ b_2/b_{exp}]$, $[\overline{c_f}/\overline{c}]$ | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $b_1/b_{exp}$ | inner spanwise flap fraction | – | $0 \leq \ldots \leq 1$ | 0 |
| $b_2/b_{exp}$ | outer spanwise flap fraction | – | $b_1/b_{exp} \leq \ldots \leq 1$ | 1 |
| $\overline{c_f}/\overline{c}$ | chordwise flap fraction | – | $0 \leq \ldots \leq 1$ | 0.25 |

This record describes flap geometry. The parameters $b_1$, $b_2$, $b_{exp}$, $\overline{c_f}$, and $\overline{c}$ are illustrated in figure 7. Note that $\overline{c}$ is sometimes defined from mid trailing edge to mid leading edge, and $\overline{c_f}$ is aligned accordingly; however, it is preferable to make them normal to the flap hinge line as shown in the figure.

The first two arguments are optional as a pair and the third argument is also optional. Thus, if only one argument is present, it is assumed to be $\overline{c_f}/\overline{c}$; if two are present, they are assumed to be $b_1/b_{exp}$ and $b_2/b_{exp}$.

| `FlaProDrag` | $k_\delta$ | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $k_\delta$ | flap deflection drag coefficient | – | $-2 \leq \ldots \leq 0$ | |

This record overrides the default value of $-0.5$ for the coefficient of profile axial force (i.e., negative drag) associated with a trailing-edge flap deflection. The coefficient is nondimensionalized with respect to exposed appendage area. See the discussion of Johnson and Hagerman's experimental data in reference [13].

| **PReverse** [*keyword*] | | | | |
|---|---|---|---|---|
| $u_{pr0}$, $u_{pr1}$, $u_{pr2}$, $u_{pr3}$ | | | | |
| [$C_{pr0}$, $C_{pr1}$, $C_{pr2}$, $C_{pr3}$] | | | | |
| **Argument(s)** | **What** | **Units** | **Range** | **Default** |
| *keyword* | **Ukt** or **Ums** | | | **Ums** |
| $u_{pr0}$, $u_{pr1}$, $u_{pr2}$, $u_{pr3}$ | speed in kt or m/s | *keyword* | see notes | |
| $C_{pr0}$, $C_{pr1}$, $C_{pr2}$, $C_{pr3}$ | corresponding control gain | – | see notes | |

The purpose of this input is to automatically compensate for the effect of plane reversal [15] under **AutoDepth** control in **Free** and **MLD** simulations and, optionally (see section 26.1), to compensate for it in an user-issued **Up** or **Down** command during a **Free** simulation. It is ignored in other commands and simulation modes.

**PReverse** input is only valid for appendages of type **Sailplane**, **Sternplane**, **Tail**, **LBowplane**, or **SBowplane**. For the last two a warning is issued that this is an "unlikely type for plane reversal".

The optional keyword argument indicates the units of speed used in the first record of the data block.

Speed data should bracket the plane reversal speed so that it lies between $u_{pr1}$ and $u_{pr2}$. If the second record is omitted, default control gains are $(-0.5, 0, 0, 1)$ for a **Sternplane** or **Tail** appendage, or $(1, 0, 0, -0.5)$ for a **Sailplane**, **LBowplane**, or **SBowplane** appendage. In all cases, speeds must be strictly increasing. To be effective, control gains should follow the same pattern as the default values: i.e., they should provide a gain of unity in the normal speed regime, negative gain outside it, and a deadband between, bracketing plane reversal.

Even if the plane reversal speed of an appendage is known *a priori*, **PReverse** input should be based on the value estimated by **DSSP50**. This value is obtained by invoking the **PReverse** option in a **Coef** simulation.

| **WeightDCI** $W_{ID}$, $W_{IH}$, [$W_{IR}$] | | | | |
|---|---|---|---|---|
| **Argument(s)** | **What** | **Units** | **Range** | **Default** |
| $W_{ID}$, $W_{IH}$ | weights for indirect depth and heading | – | $-2 \leq \ldots \leq 5$ | 1 |
| $W_{IR}$ | weight for indirect roll | – | $-2 \leq \ldots \leq 5$ | 1 |

Section 17.3 outlines the indirect method of vehicle control in which an effective deflection angle is associated with all appendages capable of controlling the vehicle in a particular mode (depth, heading, or roll, using subcommands like **Up**, **Starboard**, etc.). This method provides the mechanism for autopilot control of the vehicle. Each appendage is assigned weights that determine the degree to which it responds in each of the indirect control modes.

The weights can be used, for example, to balance bowplane and sternplane response to depth control commands, or (with a value of zero) to decouple a particular appendage from an indirect control mode. The values can be redefined in several places on file *Root*`.sim`.

## 6.4 Other Parameters

Keywords `CXJ`, `CXZero`, and `TailEff` override default values of various lifting surface parameters.

| `CXJ` $C_{Xj}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $C_{Xj}$ | junction axial force coefficient | – | $\ldots < 0$ | 0 |

The junction axial force coefficient, or junction drag, is nondimensionallized by the exposed area of the appendage.

| `CXZero` $C_{X0}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $C_{X0}$ | zero lift axial force coefficient | – | $\ldots < 0$ | $-0.01$ |

The zero lift axial force coefficient, or zero lift drag, is nondimensionallized by the exposed area of the appendage.

| `TailEff` $K_{WB}$, $[k_{WB}]$ or<br>`TailEff` *keyword* | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $K_{WB}$ | tailplane efficiency for incidence | – | $0 \leq \ldots \leq 1.5$ | |
| $k_{WB}$ | tailplane efficiency for deflection | – | $0 \leq \ldots \leq K_{WB}$ | note |
| *keyword* | `Bohlmann` or `Dempsey` | | | |

This record controls estimation of the tailplane efficiencies associated with the vehicle at incidence and with a control deflection. It must be present, with at least one numerical argument, if there is no associated hull. Bohlmann-Spreiter calculations [16] are done for tailplane appendages (`Rudder`/`Sternplane`/`Tail`) in inviscid flow and the resulting efficiency ratios $k_{WB}/K_{WB}$ are saved. In the absence of this record, tailplane efficiencies are estimated as for the `Bohlmann` keyword option.

For numerical input, the default estimate of $k_{WB}$ is obtained from the inviscid efficiency ratio, and both efficencies are constant with respect to speed.

**Bohlmann**: the efficiency ratio is ignored and viscous Bohlmann-Spreiter calculations are done for each evaluation of tailplane efficiency in a simulation. Additional corrections are made for the influence of tail cone angle and displacement thickness at small angles of incidence as discussed in reference [16].

**Dempsey**: $K_{WB}$ is calculated by Dempsey's method [14] and $k_{WB}$ is derived using the inviscid efficiency ratio; these values are then constant with respect to speed.

## 6.5 Copying Appendage Specifications

An appendage configuration can be saved, with or without itself being used, for re-use through reflection or rotation. The current appendage is first saved with **Kill** or **Save** (both overwrite a previous save), then its specifications are copied into a new location with **Reflected** or **Rotated**. Only planform corner points are modified by a copy; all other physical attributes of the appendage remain the same. However, the label is not transferred, so that a **Label** record is required for the new appendage or the default will be used; neither is the component weight (**Add**/**Subtract**/**Zero**) transferred. Multiple copies can be made of the single save.

| **Kill** |
|---|

The current appendage configuration will be saved for a subsequent copy, but will not itself be used. The user thus does not have to calculate corner points for a non-vertical or non-horizontal appendage that is more easily defined in the vehicle $xy$ or $xz$ plane and then rotated into its correct orientation.

| **Reflected** |
|---|

The saved appendage is reflected in the vehicle $xz$ plane. This record cannot be used in combination with any others in **Lift** level 2 input except for a **Label** record and a **Add**, **Subtract**, or **Zero** record.

| **Rotated** $\phi_R$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\phi_R$ | rotation angle | deg | | |

The saved appendage is rotated $\phi_R$ degrees about the vehicle $x$ axis; $\phi_R$ is positive rotating from the $y$ axis to the $z$ axis. This record cannot be used in combination with any others in **Lift** level 2 input except for a **Label** record and a **Add**, **Subtract**, or **Zero** record.

| Save |
|---|

The current appendage configuration will be first used and then saved for a subsequent copy.

# 7 Root.geo Level 2 Lump Input

A Lump is characterised by a set of hydrodynamic coefficients, not by geometry. In the program it is handled as a component, but can represent an entire vehicle, and has a full set of attributes including inertial properties and hydrodynamic control parameters.

The following table summarizes the keyword records used to characterize a Lump:

| Keyword | Function |
|---|---|
| Add | lump loads are added to the total, see section 3.1 |
| Axis | define local axis rotations about vehicle axes |
| Bowplane | introduces parameters for bowplane control |
| Coef | announces hydrodynamic coefficient values |
| Coefficnt | announces hydrodynamic coefficient values |
| File | input will be read from file |
| Label | component label, see section 3.2 |
| Length | define reference length for coefficients |
| Location | define local axis origin |
| MCB | define location of Lump CB |
| MCG | define location of Lump CG |
| MomInertia | introduces moments and cross products of inertia |
| Pop | force termination of input, see section 3.3 |
| Rudder | introduces parameters for rudder control |
| Save | input will be saved to a file |
| Sternplane | introduce parameters for sternplane control |
| Subtract | lump loads are subtracted from the total, see section 3.1 |
| Visualize | introduces plotting parameters |
| Volume | define Lump volume |
| Zero | lump loads are ignored in the total, see section 3.1 |

The default label for a lump is #LUMP*nnn* where *nnn* is the number of the lump as encountered on the input file.

A keyword record other than one of these will terminate input for the lump. After processing of level 2 data is completed, the input stream is backspaced and the new record is re-read at level 1. The records are described below in terms of the functional group that they belong to.

## 7.1 Lump Spatial and Inertial Properties

These records provide spatial or inertial information necessary to represent the lump as a component of the vehicle.

| Axis $\phi_M$, $\theta_M$, $\psi_M$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\phi_M$, $\theta_M$, $\psi_M$ | lump axis rotations | deg | | 0,0,0 |

This record defines the rotations of the local axis system with respect to the vehicle axis system. According to the usual convention, the rotations take place in reverse order.

| Length $\ell_M$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\ell_M$ | lump reference length | m | | |

This record defines the reference length with which the hydrodynamic coefficients are nondimensionalized. It is mandatory.

| Location $x_M$, $y_M$, $z_M$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $x_M$, $y_M$, $z_M$ | lump location in vehicle axes | m | | |

This record defines the origin of the lump local axis system in vehicle axes. It is mandatory.

| MCB $x_{MB}$, $y_{MB}$, $z_{MB}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $x_{MB}$, $y_{MB}$, $z_{MB}$ | lump CB in local axes | m | | 0,0,0 |

This record defines the three components of the lump center of buoyancy in local axes, i.e., with respect to its location, above.

| MCG $x_{MG}$, $y_{MG}$, $z_{MG}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $x_{MG}$, $y_{MG}$, $z_{MG}$ | lump CG in local axes | m | | 0,0,0 |

This record defines the three components of the lump center of gravity in local axes, i.e., with respect to its location, above.

| MomInertia<br>$I_{xx}$ $I_{yy}$ $I_{zz}$<br>$[I_{xy}$ $I_{xz}$ $I_{yz}]$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $I_{xx}$, $I_{yy}$, $I_{zz}$ | moments of inertia | $kg \cdot m^2$ | | 0,0,0 |
| $I_{xy}$, $I_{xz}$, $I_{yz}$ | cross products of inertia | $kg \cdot m^2$ | | 0,0,0 |

The moments and cross products of inertia are with respect to local axes and follow the usual sign convention, i.e.: $I_{xx} = \int (y^2 + z^2)\, dm$, etc., and $I_{xy} = -\int xy\, dm$, etc.

| Volume $V_M$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $V_M$ | lump volume | $m^3$ | $0 \leq \ldots$ | 0 |

Lump volume is functionally an inertial quantity because mass is derived from it assuming neutral buoyancy.

## 7.2 Lump Coefficient Model

| Coef | or | → |
|---|---|---|
| Coefficnt | | → |

This record introduces a set of hydrodynamic coefficient values at level 3. It is mandatory.

The level 3 input comprises a sequence, in any order, of records each with a keyword that is the hydrodynamic coefficient name, followed by its value × 1000. (The conventional scaling factor for hydrodynamic coefficients is 1000; it is used for both input and output.) Pop will end the sequence although it is not necessary to use it in the current build. The level 3 coefficient records can be in any order. Coefficient keywords are listed in annex B together with their equivalents in the equations of motion. If the appropriate record is omitted, a coefficient defaults to zero.

A short example of coefficient input is:

```
Coefficnt
  Xuu          -1.080
  Yvdot       -21.697
  Zwnu1q1w1    -7.433
  Xvv0         22.500
  Muuds0       -3.500
   ...
```

It is important to note that the hydrodynamic coefficients are assumed to be defined with respect to the local origin. This is particularly significant for a single lump constituting the whole vehicle since if the lump location does not coincide with the vehicle reference point some unexpected and apparently irrational results may be obtained. A similar situation will occur if the lump axes are not coincident with the vehicle reference axes. A quick check that input for a single lump is consistent is to run a **Coef** simulation; the coefficient values reported by the **LLSR** option should equal the input values. Propulsion-dependent coefficients are not estimated in the simulation, but the others are sufficient for the check.

## 7.3 Plane Deflection Control

There is an equivalency, when interpreting a command, between the bowplane, rudder, and sternplane model used for lumps and a different physical arrangement such as X-rudders. It is outlined in section 17.2.

Plane deflection control functions are specified with one of the following three records:

| **Bowplane** → |
|---|

The lump bowplane function responds to quasi-direct **Bowplane**, **Foreplane**, and **Sailplane** commands, and to indirect **Down** and **Up** commands. At least some of the associated coefficients, e.g., **Zuudb**, etc., should be nonzero.

This record is followed by the level 3 input described below.

| **Rudder** → |
|---|

The lump rudder function responds to a quasi-direct **Rudder** command, and to indirect **Port** and **Stbd/Starboard** commands. At least some of the associated coefficients, e.g., **YuudrO**, etc., should be nonzero.

This record is followed by the level 3 input described below.

| **Sternplane** → |
|---|

The lump sternplane function responds to a quasi-direct **Sternplane** command, and to indirect **Down** and **Up** commands. At least some of the associated coefficients, e.g., **MuudsO**, etc., should be nonzero.

This record is followed by the level 3 input described below.

Lump **Bowplane**, **Rudder**, and **Sternplane** control functions each take the following level 3 input (**Pop** will end the sequence):

| **Del** $\delta_\bullet$ or **Delta** $\delta_\bullet$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\delta_\bullet$ | initial control deflection | deg | | 0 |

This record defines an initial, or bias, value for control deflection. The effect of an initial value is to offset the reference deflection for some control commands.

| **DelRateLim** $\dot{\delta}_M$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\dot{\delta}_M$ | maximum control deflection rate | deg/s | $0 < \ldots$ | 9 |

In the absence of this record, the maximum rate defaults to 9 deg/s.

| **DeltaDyn** $\zeta_\delta$, $\omega_\delta$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\zeta_\delta$ | control deflection damping | – | $0 < \ldots$ | 0.9 |
| $\omega_\delta$ | control response frequency | rad/s | $0 < \ldots$ | note |

The two arguments, together with $\dot{\delta}_M$, determine control deflection time histories by way of a second order ODE for each lump control function. $\zeta_\delta$ is nondimensional; a warning is issued if it is greater than one.

In the absence of this record, $\omega_\delta$ defaults to $20\dot{\delta}_M/(\delta_{Max} - \delta_{Min})$ respectively. This is equivalent to 10 × maximum rate / |maximum deflection|.

| **DeltaLim** $\delta_{Max}$, $\delta_{Min}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\delta_{Max}$ | maximum control deflection | deg | | 35 |
| $\delta_{Min}$ | minimum control deflection | deg | $\ldots < \delta_{Max}$ | −35 |

The following level 3 record is for **Bowplane** and **Sternplane** input only:

| **PReverse** [*keyword*]<br>$u_{pr0}$, $u_{pr1}$, $u_{pr2}$, $u_{pr3}$<br>[$C_{pr0}$, $C_{pr1}$, $C_{pr2}$, $C_{pr3}$] | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *keyword* | **Ukt** or **Ums** | | | **Ums** |
| $u_{pr0}$, $u_{pr1}$, $u_{pr2}$, $u_{pr3}$ | speed in kt or m/s | *keyword* | see notes | |
| $C_{pr0}$, $C_{pr1}$, $C_{pr2}$, $C_{pr3}$ | corresponding control gain | – | see notes | |

The purpose of this input is to automatically compensate for the effect of plane reversal [15] under **AutoDepth** control in **Free** and **MLD** simulations and, optionally (see section 26.1), to compensate for it in an user-issued **Up** or **Down** command during a **Free** simulation. It is ignored in other commands and simulation modes.

The optional keyword argument indicates the units of speed used in the first record of the data block.

Speed data should bracket the plane reversal speed so that it lies between $u_{pr1}$ and $u_{pr2}$. If the second record is omitted, default control gains are $(-0.5, 0, 0, 1)$ for a **Sternplane** or **Tail** appendage, or $(1, 0, 0, -0.5)$ for a **Sailplane**, **LBowplane**, or **SBowplane** appendage. In all cases, speeds must be strictly increasing. To be effective, control gains should follow the same pattern as the default values: i.e., they should provide a gain of unity in the normal speed regime, negative gain outside it, and a deadband between, bracketing plane reversal.

Even if the plane reversal speed of a control function is known *a priori*, **PReverse** input should be based on the value derived by **DSSP50**. Knowing the coefficients, it can be calculated manually, but is more easily obtained by invoking the **PReverse** option in a **Coef** simulation.

## 7.4 Other Records

| **File** *filename*, [*keyword*] | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *filename* | lump input file name in 8.3 format | | | |
| *keyword* | **ASCII** or **Binary** | | | **Binary** |

**File** and **Save** records allow the relatively long description of a full coefficient model to be kept on a separate input file. When a **File** record is encountered, the lump input, with the exception of label, weight, and visualization data, is read from the file named in the first argument; file format is indicated by the optional second argument. The file must have been created in a previous simulation by a **Save** command, below. Having this record together with input other than the exceptions listed above generates an error.

**ASCII**: the file is a text file, i.e., in ASCII printable characters.

**Binary**: the file is in the binary format of the host operating system.

| **Save** *filename*, [*keyword*] | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *filename* | lump output file name in 8.3 format | | | |
| *keyword* | **ASCII** or **Binary** | | | **Binary** |

**File** and **Save** records allow the relatively long description of a full coefficient model to be kept on a separate input file. When a **Save** record is present, the lump specification, with the exception of label, weight, and visualization data, is written to the file named in the first argument; file format is indicated by the optional second argument. The file can be subsequently read with the **File** command.

**ASCII**: the file is a text file, i.e., in ASCII printable characters.

**Binary**: the file is in the binary format of the host OS.

| **Visualize** | **Cylinder**, *keyword*, $L_v$, $r_v$ | or | | |
|---|---|---|---|---|
| **Visualize** | **Sphere**, [$r_v$] | or | | |
| **Visualize** | **None** | | | |
| Argument(s) | What | Units | Range | Default |
| **Cylinder** | plot lump as a cylinder | | | |
| *keyword* | **X**, **Y**, or **Z** | | | |
| $L_v$ | cylinder length | m | $0 < \ldots$ | |
| $r_v$ | cylinder or sphere radius | m | $0 < \ldots$ | note |
| **Sphere** | plot lump as a sphere | | | |
| **None** | no lump plot | | | |

With the **Visualize** record, a lump is represented on the vehicle geometry plot as a cylinder or a sphere. This is primarily to check its location and, if necessary, orientation. In the absence of this record, the lump is not plotted.

**Cylinder**: the second argument indicates which of the local axes the cylinder's axis will be aligned with: $x$, $y$, or $z$. The third and fourth arguments are the length and radius of the cylinder respectively. The center of the cylinder axis is at the lump location.

**Sphere**: the optional second argument is the radius of the sphere; by default it is equal to one quarter of the lump length. The center of the sphere is at the lump location.

**None**: this is redundant in the current build.

# 8 Root.geo Level 2 OOPCalc Input

Background to the out-of-plane load calculations implemented in DSSP50 is given in reference [17]. These loads occur on a hull with asymmetric appendages, the most common case being the out-of-plane normal force and pitching moment due to the sail on a submarine. In a coefficient model, i.e., if the vehicle is represented by a lump, the loads are derived from the values of Zvv and Mvv, but in a geometry-based hydrodynamic model they are calculated as outlined in this section.

By default, the calculation is done for each appendage with the type Sail, using the nearest hull if there is more than one. The OOPCalc record and associated level 2 input are used to override default parameters, to specify calculations for other than the default condition, or to suppress the out-of-plane calculations altogether. To simply suppress them, use 'OOPCalc None'. Note that any other use of an OOPCalc record also suppresses the default calculations.

For dynamic (captive and free) simulations, the out-of-plane calculations create a time history by convecting the load distribution aft at the instantaneous value of $u$. If $u$ becomes large, there may be too few points defining the distribution to maintain precision in the load estimates, and a warning is issued:

```
?? Warning from OOPSTEP
   Depth of OOP stack less than four data points
   - precision lost in load integration
```

The user must decide whether accurate out-of-plane loads are significant for the manoeuver under consideration, and if so limit the size of the the time steps. It is generally safe to ignore this warning over short periods, especially if vehicle depth is under autopilot control.

The following table summarizes the keyword records used in OOPCalc level 2 input:

| Keyword | Function |
|---|---|
| Add | loads are added to the total — analogously to section 3.1 |
| KParam | define nonlinearity of the load distribution |
| Lambda | define the extent of the hull bound vortex |
| OHull | identify the hull component |
| OLift | identify the lift component |
| Pop | force termination of input — see section 3.3 |
| Subtract | loads are subtracted from the total — analogously to section 3.1 |
| XoneXtwo | explicitly define endpoints of the hull bound vortex |
| Zero | loads are ignored in the total — analogously to section 3.1 |

Of these records, only OLift is mandatory.

Although the loads are weighted in a similar fashion to those for hulls and other components, out-of-plane weight is internally derived from the associated hull and lift weights unless a `XoneXtwo` record is present. In the latter case an `Add`, `Subtract`, or `Zero` defines the weight if present, otherwise the `OLift` weight alone is used.

| `KParam` $k$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $k$ | load distribution parameter | $\text{rad}^{-1}$ | | 0 |

The form of the out-of-plane load distribution suggested in reference [17] consists of a linear ramp function raised to the power $m$, where $m = 1 + k|\beta_c|$, to account for nonlinearity. $\beta_c$ is the sail angle of incidence in radians. Reference [17] illustrates a number of submarine configurations for which $-2 \leq k \leq 3$. The default value represents a linear load distribution.

| `Lambda` $\lambda_1,\ \lambda_2$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\lambda_1,\ \lambda_2$ | forward and aft hull-bound vortex limits | – | see note | |

The arguments define the extent of the hull-bound vortex over which the out-of-plane load is distributed. At the forwardmost extent, $\lambda_1 = 0$, which corresponds to the sail trailing edge, and at the aftmost, $\lambda_2 = 1$, which corresponds to the hull AP. It is required that $0 \leq \lambda_1 < \lambda_2 \leq 1$. Without this record, $\lambda_1 = 0$ and $\lambda_2 = 1$. It is concluded in reference [17] that there is at present little data to justify changing these values.

| `OHull` *label* or $N_H$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *label* | label identifying the hull to be used | | | |
| $N_H$ | index identifying the hull to be used | | | |

The argument identifies the hull to be used in the calculation. Together with the `OLift` and `Lambda` records, it defines the endpoints $(x_1, x_2)$ of the hull-bound vortex.

Alternatively, $(x_1, x_2)$ can be input explicitly with the `XoneXtwo` record. If neither record is present, the hull nearest to the `OLift` component is used for the calculation.

| `OLift` *label* or $N_L$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *label* | label identifying the appendage to be used | | | |
| $N_L$ | index identifying the appendage to be used | | | |

The argument identifies the lifting appendage (which may be of any type) to be used in the calculation.

**XoneXtwo**

$x_{1x}$ $x_{1y}$ $x_{1z}$

$x_{2x}$ $x_{2y}$ $x_{2z}$

| Argument(s) | What | Units | Range | Default |
|---|---|---|---|---|
| $x_{1x}$, $x_{1y}$, $x_{1z}$ | forward hull-bound vortex endpoint | m | | |
| $x_{2x}$, $x_{2y}$, $x_{2z}$ | aft hull-bound vortex endpoint | m | | |

The hull-bound vortex endpoints are defined in vehicle coordinates; $(x_1, x_2)$ correspond to $(\lambda_1, \lambda_2)$.

Only one, or the other, or neither, of this and the `OHull` records can be present. If neither is present, the hull nearest to the `OLift` component is used in the calculation.

There is no check that these data correspond to an actual vehicle hull.

# 9 Root.geo Level 2 Propulsor Input

Propulsor types are discussed above, in section 4.3. The input described in this section is applicable to all of them.

The following table summarizes the keyword records used to characterize a generalized propulsor:

| Keyword | Function |
|---|---|
| `Diameter` | define propulsor diameter |
| `Label` | propulsor label, see section 3.2 |
| `Left` | left-handed propulsor |
| `Location` | define propulsor local origin |
| `Pitch` | define local axis pitch |
| `Pop` | force termination of input — see section 3.3 |
| `Reflected` | copy saved propulsor by reflecting |
| `RevDynam` | set RPM dynamic parameters |
| `RevGroups` | define propulsor switching groups |
| `RevLimit` | set maximum RPM |
| `RevRateLim` | set RPM change rate limit |
| `Right` | right-handed propulsor |
| `Save` | save this propulsor for subsequent copy |
| `Yaw` | define local axis yaw |

The **Label** definition is analogous to that of other components. If there is no **Label** record, the default label is **#PROP***nnn* where *nnn* is the number of this propulsor as encountered in the input file.

## 9.1 Propulsor Geometry

| **Diameter** $D_P$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $D_P$ | propulsor diameter | m | $0 \leq \ldots$ | |

This record is mandatory.

| **Location** $x_P$, $[y_P, z_P]$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $x_P$ | $x$ component of local origin | m | | |
| $y_P$, $z_P$ | $y$ and $z$ components of local origin | m | | 0,0 |

The propulsor local origin, which is defined in vehicle axes, is the point at which estimated thrust and torque are applied (it is assumed that the points of application of these loads are colocated). The distinction between these points of application and the propulsor's physical location may have to be considered, but for most purposes it is likely sufficient to use the location of the propulsor hub.

This record is mandatory.

| **Pitch** $\theta_P$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\theta_P$ | pitch of propulsor axes | deg | | 0 |

This record defines the pitch angle of the propulsor axis with respect to vehicle axes. The order of rotation is $\psi_P$ (below), $\theta_P$.

| **Yaw** $\psi_P$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\psi_P$ | yaw of propulsor axes | deg | | 0 |

This record defines the yaw angle of the propulsor axis with respect to vehicle axes. The order of rotation is $\psi_P$, $\theta_P$ (above).

## 9.2 Other Propulsor Parameters

| `Left` |
|---|

The propulsor is left-handed. For self-propulsion ahead, the propulsor has counterclockwise rotation looking forward, so that torque on the vehicle is positive. If this record is omitted, the propulsor is right-handed.

| `RevDynam` $\zeta_n$, $\omega_n$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\zeta_n$ | change-of-RPM damping | – | $0 < \ldots$ | 0.9 |
| $\omega_n$ | change-of-RPM frequency | rad/s | $0 < \ldots$ | note |

The two arguments, together with $\dot{n_M}$ (below), determine change-of-RPM time histories by way of a second order ODE for each propulsor. $\zeta_n$ is nondimensional; a warning is issued if it is greater than one.

These parameters are the same for all propulsion groups. In the absence of this record, $\omega_n$ defaults to $10\dot{n}_M/n_M$.

| `RevGroups` $[n_M]$ $\begin{bmatrix} n_{sw1} & t_{sw1} \\ n_{sw2} & t_{sw2} \\ \text{etc.} & \end{bmatrix}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $n_M$ | maximum RPM | RPM | $0 < \ldots$ | note |
| $n_{swi}$ | upper RPM boundary of the $i^{th}$ group | RPM | $0 < \ldots$ | |
| $t_{swi}$ | delay at lower RPM boundary of the $i^{th}$ group | sec | | |

`RevGroups` approximates the time delays incurred in switching between battery groups in an electric propulsion system. If neither this record nor a `RevLimit` record is present, maximum RPM is $926/D_P$ (for $D_P$ in m), see `RevLimit`.

This record must have either the single argument $n_M$ or the data block specifying propulsion groups. With the single argument, it is equivalent to a `RevLimit` record. With the data block, $n_M$ is equal to $n_{swi}$ on the last record of the block.

The data block must be organized so that $n_{swi}$ is monotonically increasing. Only the ahead groups are defined; the astern groups are assumed to be symmetrical with them. The lower boundary of the first group is taken to be zero, and the lower boundary of subsequent ones is equal to the upper boundary of the one previous.

When the lower boundary of a group is crossed, RPM is held constant for the time delay indicated.

| **RevLimit** $n_M$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $n_M$ | maximum RPM | RPM | $0 < \ldots$ | note |

If neither this record nor a **RevGroups** record is present, maximum RPM is $926/D_P$ (for $D_P$ in m), i.e., RPM for an advance ratio of one at 30 kt.

| **RevRateLim** $\dot{n}_M$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\dot{n}_M$ | maximum change-of-RPM rate | RPM/s | $0 < \ldots$ | 10 |

This parameter is the same for all propulsion groups.

**Right**

The propulsor is right-handed. For self-propulsion ahead, the propulsor has clockwise rotation looking forward, so that torque on the vehicle is negative. This record is redundant since right-handedness is the default.

## 9.3 Saving and Copying

Similarly to lifting appendages (section 6.5), a propulsor may be saved and copied; however, only **Save** and **Reflected** records are applicable. The propulsor label is not saved, while all other attributes are.

**Reflected**

The saved propulsor is reflected in the vehicle $xz$ plane. A **Label** record is required for the new propulsor or the default **#PROP***nnn* will be used. This record cannot be used with any other level 2 propulsor input apart from **Label**. Note that handedness (left/right) is reversed in a reflection.

**Save**

The current propulsor will be used and then saved for a subsequent copy. This overwrites a previous save.

# 10 Root.geo Level 2 WagBInit Input

Input discussed in this section is used to initialize propulsors with type **WageningB** (section 4.3): i.e., Wageningen B4–70 series propellers [10]. If this type is used, all propulsors must be the same type. The initialization estimates optimum $J_S$ and RPM/kt values for given operational conditions.

Wageningen B4–70 propeller thrust and torque characteristics are derived from lookup tables taken from table 7 in reference [10]. It is assumed that all propellers contribute to forward thrust; for initialization, the total thrust is apportioned between them in a specified ratio. The same form of calculation is used for all, so the program does not accurately handle the situation of a main propulsor of this type on the hull axis with an auxiliary propulsor on each sternplane (i.e., effectively in open water).

The following table summarizes the keyword records for level 2 **WagBInit** input.

| Keyword | Function |
|---|---|
| **Diameter** | define a hull reference diameter |
| **Open** | use open water propeller option |
| **PoD** | set pitch/diameter ratio |
| **Pop** | force termination of input — see section 3.4 |
| **Rho** | set fluid density for propeller initialization |
| **TRatio** | set thrust ratios for multiple propellers |
| **Ukt** | ahead speed for propeller initialization, in kt |
| **Ums** | ahead speed for propeller initialization, in m/s |

**Diameter** $D_M$

| Argument(s) | What | Units | Range | Default |
|---|---|---|---|---|
| $D_M$ | maximum hull diameter | m | $0 < \ldots$ | |

The maximum diameter of the hull is used to form the ratio $D_P/D_M$ for estimating wake fraction and thrust deduction coefficients. This record may be omitted if the vehicle has only one hull, in which case $D_M$ defaults to its calculated maximum diameter. If there is no hull, or more than one hull, a value for $D_M$ must be input explicitly.

**Open**

This record forces the open water condition of wake fraction and thrust deduction equal to zero, and may be appropriate for using **WageningB** propellers in the old twin-screw submarine arrangement. In its absence, the default is to estimate wake fraction and thrust deduction for propellers in the wake of a coaxial hull.

| PoD $P_P/D_P$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $P_P/D_P$ | propeller pitch/diameter ratio | – | $0.6 \leq \ldots \leq 1.4$ | |

This record overrides the default estimate of optimum pitch/diameter ratio for **WageningB** propellers; the same value is used for all. If the ratio is input or calculated to be in the range $0.5 \leq P_P/D_P < 0.6$, it is simply reset to 0.6. If it is less than 0.5 or greater than 1.4, the user is given the option (i.e., warning with pause) to continue using the limiting value of 0.6 or 1.4 respectively, or to terminate the program.

| Rho $\rho$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\rho$ | water density | $\mathrm{kg/m^3}$ | $990 \leq \ldots \leq 1035$ | |

This record overrides the default water density value of 1000 for **WageningB** propeller initialization. It does not permanently reset $\rho$.

| TRatio $\tilde{T}_i,\ i = 1, \ldots$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\tilde{T}_i,\ i = 1, \ldots$ | individual propeller thrust | see note | | |

For initializing multiple **WageningB** propellers, the proportion, $\tilde{T}_i$, of thrust developed by each one can be specified. There must be one value of $\tilde{T}_i$ for each propeller. They will be normalized by their total, $\sum_i \tilde{T}_i$, and can be in any convenient form such as estimated thrust values or percentages. If this record is absent, $\tilde{T}_i$ defaults to the disk area of the $i^{\text{th}}$ propeller, so that each one is nominally equally loaded.

| Ukt $U_\bullet$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $U_\bullet$ | vehicle speed for initialization | kt | | |

This record overrides the default value of 5 kt for vehicle speed that is used to initialize advance ratio and RPM/kt for **WageningB** propellers. There can be only one **Ukt** or **Ums** record.

| Ums $U_\bullet$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $U_\bullet$ | vehicle speed for initialization | m/s | | |

This record overrides the default value of 2.57222 m/s (= 5 kt) for vehicle speed that is used to initialize advance ratio and RPM/kt for `WageningB` propellers. There can be only one `Ums` or `Ukt` record.

## 11 Root.geo Level 2 Autopilot Input

The records described here follow an `AutoDepth`, `AutoHead`, `AutoProp`, or `AutoRoll` record. First, to understand level 2 autopilot input, a brief outline of the control algorithms will be useful.

The error signal $J_E$, for a quantity $Q$ (i.e., heading, roll, etc) is in general:

$$J_E = [Q_c - Q]\, C_E \tag{1}$$

where $Q_c$ is the commanded value of $Q$, and $C_E$ (in units of $Q^{-1}$) is an error gain required to normalize $J_E$ to within $-1 \geq J_E \geq 1$ for a typical maximum response. Such definitions are deliberately vague; some trial and error will be required to find optimum values for the gains and other autopilot parameters.

For propulsion control, the error signal has the form $[Q_c^n - Q^n]\, C_E$, where $n = 1/3$ if $Q$ is power, $n = 1$ if $Q$ is speed, or $n = 1/2$ if $Q$ is thrust, and the units of $C_E$ correspond in each case.

The depth error signal is also a little more complex than equation (1) as it depends on pitch angle, $\theta$, in addition to depth, $z_0$:

$$J_{E\,(\text{depth})} = \left[ \left[ z_{0c} - z_0 \right]_{-l_z \sin\theta_L}^{+l_z \sin\theta_L} + l_z \sin\theta \right] C_{E\,(\text{depth})} \tag{2}$$

where $l_z$ is the lookahead distance (making it appear that depth is sensed $l_z$ ahead of the coordinate origin), and $\theta_L$ is the autopilot pitch limit. In formulating $J_{E\,(\text{depth})}$, the full equation (2) is used by default, although use of the depth or pitch contributions alone can be invoked during a manoeuvering simulation, section 26.2.

In all cases, $J_E$ is clipped at $\pm 1$. There are two methods in `DSSP50` for deriving the corresponding control signal $J_C$. The first is a phase lead algorithm:

$$\dot{J}_C = \frac{\alpha}{\tau} \left[ J_E + \tau \dot{J}_E - J_C \right] \tag{3}$$

where $\alpha$ is the amplitude gain of the transformation and $\tau$ is the anticipation time constant [18]. After rearrangement, equation (3) is solved as a first order ODE in the time integration.

The second method for deriving $J_C$ is a PID (Proportional-Integral-Derivative) algorithm:

$$J_C = K_P\, J_E + K_I \int J_E\, dt + K_D\, \dot{J}_E \tag{4}$$

where $K_P, K_I, K_D$ are the PID coefficients. To solve this in the time integration, a lightly damped error signal $J'_E$ is used in the derivative term.

$$k_1 \dot{J}'_E + k_2 J'_E = J_E \tag{5}$$

The control signal is then obtained from two first order ODEs using both the damping equation coefficients, $k_1$ and $k_2$, and the PID coefficients.

$J_C$ is clipped at $\pm 1$ and multiplied by a control gain $C_C$ in appropriate units (degrees for control deflections, RPM for propulsion revolutions) to determine the control correction. Time histories of $J_C$ and $J_E$ for the various autopilots are tabulated in the auxiliary output file `aux`*nnn*`.txt`.

Defaults are provided for all the parameters discussed above, but optimizing them will require some trial and error depending on the geometric and other characteristics of the vehicle. None of the records listed in sections 11.1 to 11.5 is mandatory. However, if a record is input, all the arguments for it must be included.

## 11.1 Deriving Control Signals

The `PhaseLead` and `PID` records define type for any of the four autopilots. For each autopilot there can be only one or neither of these two records. If there is neither, phase lead is the default type for the depth, heading, or roll autopilots and PID is the default type for the propulsion autopilot.

| `PhaseLead` $\alpha$, $\tau$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\alpha$ | amplitude gain | – | | 3 |
| $\tau$ | anticipation time constant | s | | 2 |

A `PhaseLead` record is required to change any of the default parameter values.

| `PID` $k_1$, $k_2$, $K_P$, $K_I$, $K_D$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $k_1$ | error signal damping time constant | s | | 0.5 |
| $k_2$ | error signal damping gain | – | | 1 |
| $K_P$ | proportional coefficient in PID equation | – | | 1 |
| $K_I$ | integral coefficient in PID equation | $s^{-1}$ | | 0 |
| $K_D$ | derivative coefficient in PID equation | s | | 0 |

Note that the default is proportional control, although the error signal is still damped. A `PID` record is required to change any of the default parameter values.

## 11.2 AutoDepth Level 2 Input

The following table summarizes the keyword records for **AutoDepth** level 2 input.

| Keyword | Function |
|---|---|
| **DepConGain** | set depth control gain |
| **DepErrGain** | set depth error gain |
| **LookAhead** | set look ahead parameter |
| **PhaseLead** | phase lead control, see section 11.1 |
| **PID** | PID control, see section 11.1 |
| **PitchLimit** | set autopilot pitch limit |
| **Pop** | force termination of input — see section 3.3 |

| **DepConGain** $C_C$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $C_C$ | depth control gain | deg | $0 < \ldots$ | 20 |

Since the commanded deflection depends on appendage dihedral and control weight $W_{ID}$ (section 6.3), this parameter can exceed the maximum deflection on an appendage.

| **DepErrGain** $C_E$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $C_E$ | depth error gain | $\mathrm{m}^{-1}$ | $0 < \ldots$ | note |

Without this record, and with just one hull of length $\ell$, the depth error gain is $12/\ell$, which assumes a look ahead of $\ell/2$ and the default pitch limit of 10 degrees. Otherwise, without this record $C_E$ is zeroed and a warning with pause is issued.

| **LookAhead** $l_z$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $l_z$ | look ahead distance | m | | note |

Without this record, and with just one hull of length $\ell$, the look ahead is set to $\ell/2$ Otherwise, without this record $l_z$ is zeroed and a warning with pause is issued.

| **PitchLimit** $\theta_L$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\theta_L$ | depth autopilot pitch limit | deg | $0 < \ldots$ | 10 |

The autopilot will try to avoid exceeding this limit (either positive or negative) at the expense of depth control.

## 11.3 AutoHead Level 2 Input

The following table summarizes the keyword records for `AutoHead` level 2 input.

| Keyword | Function |
|---|---|
| `HedConGain` | set heading control gain |
| `HedErrGain` | set heading error gain |
| `PhaseLead` | phase lead control, see section 11.1 |
| `PID` | PID control, see section 11.1 |
| `Pop` | force termination of input — see section 3.3 |

| `HedConGain` $C_C$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $C_C$ | heading control gain | deg | $0 < \ldots$ | 20 |

Since the commanded deflection depends on appendage dihedral and control weight $W_{IH}$ (section 6.3), this parameter can exceed the maximum deflection on an appendage.

| `HedErrGain` $C_E$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $C_E$ | heading error gain | $\text{deg}^{-1}$ | $0 < \ldots$ | 0.05 |

Without this record, the heading error gain is $0.05\,\text{deg}^{-1}$.

## 11.4 AutoProp Level 2 Input

The following table summarizes the keyword records for `AutoProp` level 2 input.

| Keyword | Function |
|---|---|
| `PhaseLead` | phase lead control, see section 11.1 |
| `PID` | PID control, see section 11.1 |
| `Pop` | force termination of input — see section 3.3 |
| `PwrErrGain` | set power error gain |
| `RevConGain` | set revolutions control gain |
| `ThrErrGain` | set thrust error gain |
| `UktErrGain` | set speed error gain (for kt) |
| `UmsErrGain` | set speed error gain (for m/s) |

| PwrErrGain $C_{E\,(\text{power})}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $C_{E\,(\text{power})}$ | power error gain | $\text{W}^{-1/3}$ | $0 < \ldots$ | 0.12 |

The default value is based on a rough estimate of power required for a small submarine at about 1 kt.

| RevConGain $C_C$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $C_C$ | propulsor revolutions control gain | RPM | $0 < \ldots$ | 12 |

The default value is an approximation based on a speed of 1 kt.

| ThrErrGain $C_{E\,(\text{thrust})}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $C_{E\,(\text{thrust})}$ | thrust error gain | $\text{N}^{-1/2}$ | $0 < \ldots$ | 0.0375 |

The default value is based on a rough estimate of thrust required for a small submarine at about 1 kt.

| UktErrGain $C_{E\,(\text{speed})}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $C_{E\,(\text{speed})}$ | speed error gain | $\text{kt}^{-1}$ | $0 < \ldots$ | 1 |

This record is used to input the speed error gain in $\text{kt}^{-1}$.

| UmsErrGain $C_{E\,(\text{speed})}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $C_{E\,(\text{speed})}$ | speed error gain | $(\text{m/s})^{-1}$ | $0 < \ldots$ | note |

This record is used to input the speed error gain in $(\text{m/s})^{-1}$. Without it, the default value is $1/0.514\dot{4}$ (i.e., $1\ \text{kt}^{-1}$).

## 11.5 AutoRoll Level 2 Input

The following table summarizes the keyword records for `AutoRoll` level 2 input.

| Keyword | Function |
|---|---|
| `PhaseLead` | phase lead control, see section 11.1 |
| `PID` | PID control, see section 11.1 |
| `Pop` | force termination of input — see section 3.3 |
| `RolConGain` | set roll control gain |
| `RolErrGain` | set roll error gain |

| `RolConGain` $C_C$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $C_C$ | roll control gain | deg | $0 < \ldots$ | 20 |

Since the commanded deflection depends on appendage dihedral and control weight $W_{IR}$ (section 6.3), this parameter can exceed the maximum deflection of an appendage.

| `RolErrGain` $C_E$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $C_E$ | roll error gain | $\text{deg}^{-1}$ | $0 < \ldots$ | 0.05 |

Without this record, the error gain is 0.05.

## 12 Root.geo Level 2 Compress Input

Compression data can be supplied for the pressure hull, for resilient tiles cladding the vessel, or both. In each case, the data specify loss of buoyant mass and effective location at a series of increasing depths. At intermediate depths, the program does piecewise linear interpolation on the data. At depths greater than the maximum in the data, the hull compression calculation extrapolates the last data segment, whereas tile compression uses the last data values given (i.e., there is no further tile compression). Both calculations assume zero loss of buoyancy at zero depth, but location for zero depth is set to the value in the first data record.

The keywords for hull and tile compression are `CHull` and `Tile` respectively; there is no associated compression if either is absent. `Pop` can be used to exit.

| CHull | | | | |
|---|---|---|---|---|
| | $z_{ch1}$ $B_{ch1}$ $x_{ch1}$<br>$z_{ch2}$ $B_{ch2}$ $x_{ch2}$<br>etc. | | | |
| Argument(s) | What | Units | Range | Default |
| $z_{chi}$ | depth | m | $0 < z_{ch1};\ z_{ch1} < z_{chi+1}$ | |
| $B_{chi}$ | loss of buoyant mass | kg | $0 < B_{ch1};\ B_{ch1} < B_{chi+1}$ | |
| $x_{chi}$ | effective longitudinal location | m | | |

There can be up to ten records in the data block; $0 < i \leq 10$. Location is relative to the vehicle reference.

| Tile | | | | |
|---|---|---|---|---|
| | $z_{ct1}$ $B_{ct1}$ $x_{ct1}$<br>$z_{ct2}$ $B_{ct2}$ $x_{ct2}$<br>etc. | | | |
| Argument(s) | What | Units | Range | Default |
| $z_{cti}$ | depth | m | $0 < z_{ct1};\ z_{ct1} < z_{cti+1}$ | |
| $B_{cti}$ | loss of buoyant mass | kg | $0 < B_{ct1};\ B_{ct1} < B_{cti+1}$ | |
| $x_{cti}$ | effective longitudinal location | m | | |

There can be up to ten records in the data block; $0 < i \leq 10$. Location is relative to the vehicle reference.

## 13 Root.geo Level 2 HPAir and MBT Input

The blowing model in DSSP50 allows representation of a number of typical water ballast system configurations. There can be up to nine HP air bottle groups (BGs) and eight main ballast tanks (MBTs) in the system. Figure 8 illustrates an arrangement of three BGs and three MBTs in which one of the BGs is dedicated to a single MBT, one can be used with two of them, and the third can be used with any of the three. Isolation of the HP air sources from each other by check valves, as shown in the figure, is assumed. A similar arrangement to figure 8 is found in some submarine systems that employ a main HP blowing source, which in the figure would be BG2, with additional augmentation or emergency sources, BG1 and BG3. In some cases, tanks can only be blown together, for example if MBT1 and MBT2 were connected to BG1 with a single gate valve instead of separate ones. In DSSP50, this distinction is not made in the definition of the components, but can be made with Blow commands, section 19. Small UUVs typically have simpler systems comprising either a single source or an individual source integrated with each tank.

The connections are modeled in the program by associations between BGs and MBTs. Associations are set up by assigning a type to the components or are otherwise established by default. (Although only one BG of each type is usually necessary to characterize a ballast system, `DSSP50` allows more that one BG of each type.) The rules for association are:

- a BG of type `Main` is associated with all MBTs;
- a BG of type `Forward` or `Aft` is associated with all tanks of the same type; and
- remaining BGs are paired off with remaining MBTs in the order in which they are specified on the *Root*`.geo` file.

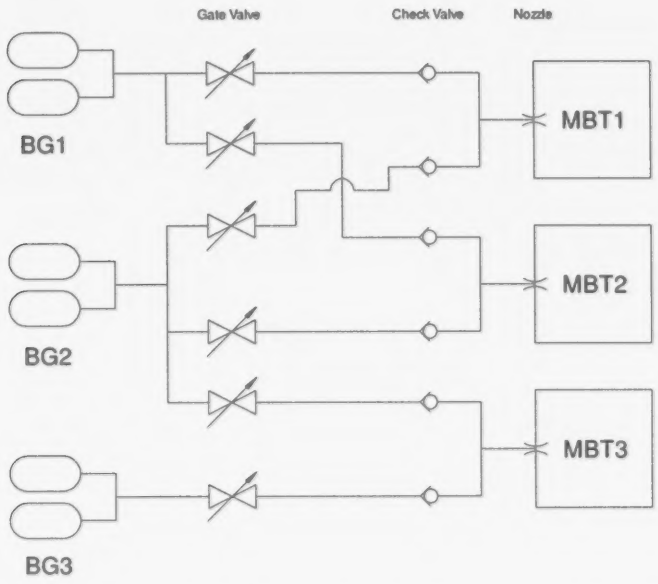


**Figure 8.** Schematic of connections between three bottle groups and three ballast tanks. BG1 can blow MBT1 and MBT2 individually or together, BG2 can blow any of the three tanks individually or together, and BG3 can blow only MBT3.

For example, the connections of figure 8 can be established by giving BG1, MBT1, and MBT2 type `Forward`, giving BG2 type `Main`, and giving BG3 and MBT3 type `Aft`.

Blowing a tank with more than one BG is computationally inefficient in the current blowing model. If the simulation consists of a single blow of this type, combining the BGs to be used into a single source of the same total volume will avoid the problem. The pressure of the single source, $P_{total}$, can be obtained from $P_{total}V_{total} = \sum_i P_i V_i$, where $i$ runs over the BGs used. Isothermal and adiabatic blowing are discussed in

section 19; the combined BG model gives identical results to blowing with individual BGs isothermally, and quite similar results to blowing with them adiabatically unless the initial pressures are very different.

## 13.1 HPAir Level 2 Input

A bottle group is announced by a `HPAir` record. The following table summarizes the level 2 input that characterizes it:

| Keyword | Function |
|---|---|
| `Aft` | set BG type |
| `Forward` | set BG type |
| `Fwd` | set BG type |
| `Label` | BG label |
| `Main` | set BG type |
| `Pop` | force termination of input — see section 3.3 |
| `Pressure` | set BG initial pressure |
| `Volume` | set BG volume |

The `Label` definition is analogous to that of other components. If there is no `Label` record, the default label is `#HPBG`*nnn* where *nnn* is the number of this BG as encountered in the input file.

```
Aft
```

This record sets the BG to type `Aft`. It will be associated with MBTs of type `Aft`.

```
Forward      or
Fwd
```

This record sets the BG to type `Forward`. It will be associated with MBTs of type `Forward`.

```
Main
```

This record sets the BG to type `Main`. It will be associated with all MBTs.

| Pressure $P_{HP\bullet}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $P_{HP\bullet}$ | initial BG pressure | bar | $100 \leq \ldots \leq 400$ | |

This record is mandatory.

| Volume $V_{HP}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $V_{HP}$ | BG volume | $m^3$ | $0 < \ldots$ | |

This record is mandatory.

## 13.2 MBT Level 2 Input

A ballast tank is announced by a `MBT` record. The following table summarizes the level 2 input that characterizes it:

| Keyword | Function |
|---|---|
| `Aft` | set MBT type |
| `Forward` | set MBT type |
| `Fwd` | set MBT type |
| `Label` | MBT label |
| `Location` | set location of MBT centroid |
| `MassFlow` | set initial blowing mass flow into MBT |
| `Pop` | force termination of input — see section 3.3 |
| `Vent` | set air mass flow venting rate |
| `Volume` | set MBT blowable volume |

The `Label` definition is analogous to that of other components. If there is no `Label` record, the default label is `#BALT`*nnn* where *nnn* is the number of this MBT as encountered in the input file.

`Aft`

This record sets the MBT to type `Aft`. It will be associated with BGs of type `Aft` (and `Main`).

`Forward` or
`Fwd`

This record sets the MBT to type **Forward**. It will be associated with BGs of type **Forward** (and **Main**).

| **Location** $x_{BT}$, $y_{BT}$, $z_{BT}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $x_{BT}$, $y_{BT}$, $z_{BT}$ | MBT centroid in vehicle axes | m | | |

This record is mandatory. In the present build, changes of mass that result from blowing or venting are assumed to occur at this point.

| **MassFlow** $\dot{m}_{BT\bullet}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\dot{m}_{BT\bullet}$ | initial blowing mass flow rate | kg/s | | |

This record is mandatory. It defines the initial blowing mass flow rate of air into the MBT. At typical HP air pressures, this quantity is determined by the nozzle that releases air into the tank.

| **Vent** $\dot{m}_{BV}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $\dot{m}_{BV}$ | venting mass flow rate | kg/s | | 0 |

This record defines the venting mass flow rate of air from the MBT. The rate is constant for the duration of the simulation, independent of depth. In the absence of this record there is no venting.

| **Volume** $V_{BT}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $V_{BT}$ | blowable volume of the MBT | $m^3$ | | |

This record is mandatory.

## 14 Root.geo Level 2 WTC Input

There can be up to eight watertight compartments (WTCs) defined in the vehicle; they are required for modeling a flood during a **Free** or **MLD** simulation. Each compartment is announced by a **WTC** record and defined by subsequent level 2 input.

A WTC is assumed to be a circular cylinder characterized by diameter, length, the location of its center, and occupancy. The moment due to floodwater is estimated as a function of both flooded volume and pitch angle.

The following table summarizes `WTC` level 2 input:

| Keyword | Function |
|---|---|
| `Diameter` | set WTC diameter |
| `Label` | WTC label |
| `Length` | set WTC length |
| `Location` | set location of WTC center point |
| `Occupancy` | set WTC occupancy |
| `Pop` | force termination of input — see section 3.3 |

The `Label` definition is analogous to that of other components. If there is no `Label` record, the default label is `#WTCP`*nnn* where *nnn* is the number of this WTC as encountered in the input file.

| `Diameter` $D_C$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $D_C$ | characteristic compartment diameter | m | $0 < \ldots$ | |

This record is mandatory. Note that the diameter is not checked for consistency with any other input (such as a hull diameter).

| `Length` $L_C$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $L_C$ | characteristic compartment length | m | $0 < \ldots$ | |

This record is mandatory. Note that the length is not checked for consistency with any other input (such as a hull length).

| `Location` $x_C$, $y_C$, $z_C$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $x_C$, $y_C$, $z_C$ | centroid of the WTC in vehicle axes | m | | |

This record is mandatory. It defines the coordinates of the centroid of the unoccupied cylinder modeling the WTC. Note that this point is not checked for consistency with any other input (such as hull location and dimensions).

| Occupancy $O_C$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $O_C$ | occupancy of the WTC | – | $0.05 \leq \ldots \leq 0.95$ | 0.3 |

Occupancy is the fraction of compartment volume that is taken up by machinery, etc., and therefore not available for flooding. The maximum amount of floodwater that can enter is $\frac{\pi}{4} D_C{}^2 L_C (1 - O_C)$.

## 15 Root.geo Level 2 Comp, Trim, and TrimComp Input

Compensation and trim is treated as one system. The tanks in the system are associated with one or both of these functions according to which keyword is used to introduce them: `Comp`, `Trim`, or `TrimComp`. For compensation, water is transferred between a tank and the sea, whereas for trim, water is transferred from one tank to another. These transfers take place at `CompRate` or `TrimRate` defined in *Root*`.geo`.

`Comp`, `Trim`, and `TrimComp` records each take the following level 2 input:

| Keyword | Function |
|---|---|
| `Label` | tank label |
| `Location` | set location of tank centroid |
| `Pop` | force termination of input — see section 3.3 |
| `Volume` | set tank volume |

The `Label` definition is analogous to that of other components. If there is no `Label` record, the default label is `#TCMP`*nnn* where *nnn* is the number of this tank as encountered in the input file.

| Location $x_{tct}$, $y_{tct}$, $z_{tct}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $x_{tct}$, $y_{tct}$, $z_{tct}$ | centroid of the empty tank volume | m | | |

This record is mandatory. The point is not checked for consistency with any other input such as hull geometry.

| Volume $V_{tct}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $V_{tct}$ | empty volume of the tank | $m^3$ | | |

A `Volume` record is mandatory.

## 16 Root.geo Level 2 Plot Input

If a `Plot` record is found at level 1, plotting output of the vehicle geometry is saved in ASCII characters on file *Root*`.gpl` (the file extension may depend on the target plotting software). A plot enables the user to check that the geometry has been input correctly. Component intersections should be checked with a wire frame plot using no hidden lines so that the details are not obscured.

Plotting requires representation of the component cross sections, which are artefacts not defined or used elsewhere in the program. Hull cross sections are represented by an ellipse fitted to $B_i$ and $T_i$; the ellipse has twenty four perimeter segments. Lifting component cross sections, at the root and tip only, are represented by a NACA 4-digit section fitted to their chord and maximum thickness; the section has eight perimeter segments.

Lumps are omitted by default but can be plotted as a cylinder or a sphere according to the `Visualize` record described in section 7.4. A cylinder has twenty four girthwise segments; a sphere has twenty four around the parallels and twelve between poles.

The format of *Root*`.gpl` is controlled by providing one of the following optional input records at level 2:

| Keyword | Function |
|---|---|
| `AcroSpin` | output *Root*`.acd` in Acrospin format |
| `Maple` | format *Root*`.gpl` for the Maple procedure `DSSPplot.mpl` |
| `MatLab` | format *Root*`.gpl` for the MATLAB script `DSSPplot.m` |
| `VRML` | output *Root*`.wrl` in VRML 1 format |

If none of these records is found, the default file format is VRML 1.

`AcroSpin`

The plot is formatted for AcroSpin Version 2.0 (DOS) in a file named *Root*`.acd`. This is a legacy option; the program appears to be no longer available. AcroSpin produces a wire-frame screen plot with fast zoom, rotation, and translation capabilities. An example screen shot is shown in figure 9.

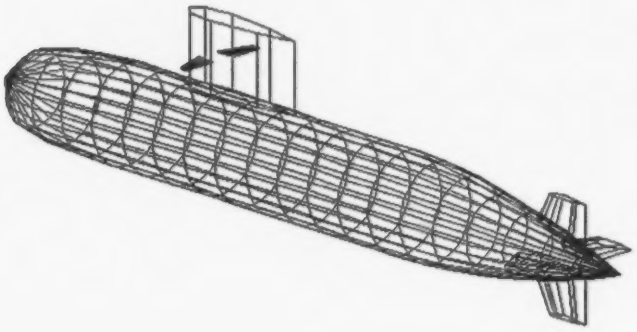

**Figure 9.** `AcroSpin` plot screen shot.

## Maple

*Root*`.gpl` is formatted for a Maple (`http://www.maplesoft.com`) procedure written by George Watt at DRDC. This procedure, `DSSPplot.mpl`, has not been tested on recent versions of Maple. Options include solid modeling (patch plot) or wire frame with zoom, rotation, translation, and perspective capabilities. A basic patch plot is shown in figure 10.

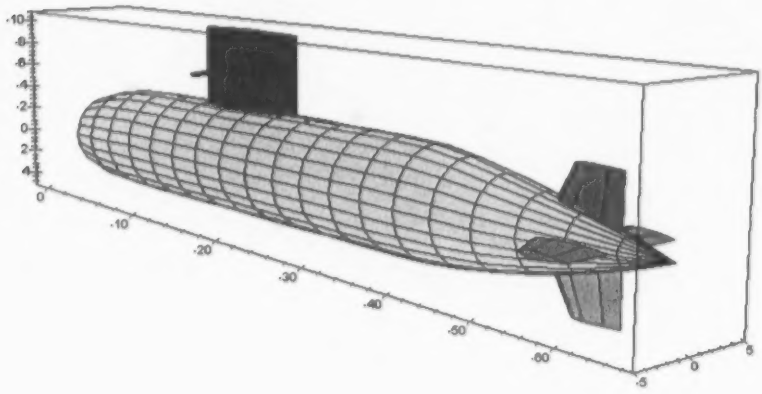


**Figure 10.** `Maple` plot screen shot; component colour is related to its input order, not type. The axis box is optional.

| MatLab | *[keyword]* | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *keyword* | `Colour` or `White` | | | `White` |

*Root*`.gpl` is formatted for a MATLAB (`http://www.mathworks.com`) script written by the author. This script, `DSSPplot.m`, has not been tested on recent versions of MATLAB.

`Colour`: the result is a patch plot with patches that are yellow on hull components, cyan on non-controlling appendages, red on controlling appendages, and green on lumps.

`White`: the patches are white and the standard lighted view appears in greyscale. The white patch image also permits a wireframe view by the user invoking 'hidden off' at the MATLAB command line: hidden line removal is turned off and the patches are made transparent. This does not work with coloured patches.

To examine the image it is easiest to invoke 'Camera Toolbar' from the figure window View menu, and then use the camera controls to orient the image and to adjust perspective, viewpoint, etc.

Example screen-shots are shown in figures 11 and 12.

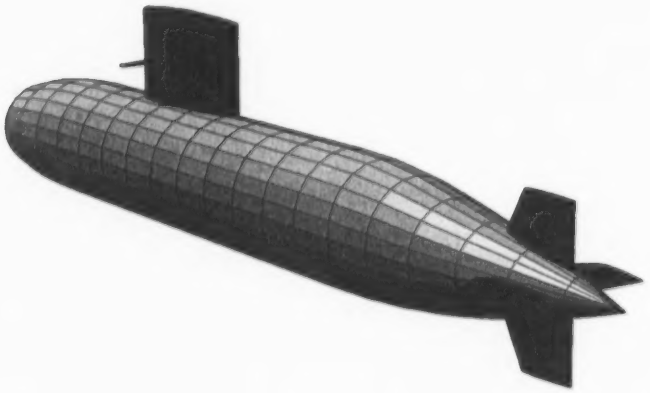

**Figure 11.** `MatLab` plot screen shot produced using the keyword `Colour`; the axis box is hidden.

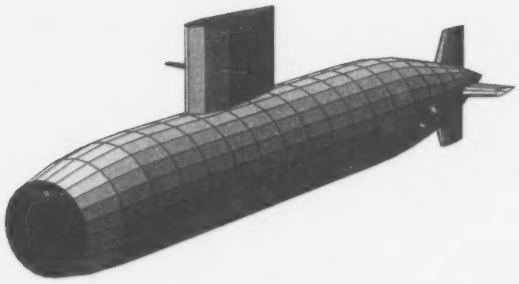

**Figure 12.** `MatLab` plot screen shot produced using the (default) keyword `White` and additional external lighting: red to port, green to starboard, yellow above, and blue below.

`VRML`

The plot is output in VRML 1 format; the plotting file is named *Root*`.wrl`. Components are coloured: hulls are yellow, non-controlling appendages are cyan, controlling appendages are red, and lumps are green.

VRML was developed in the mid-1990s as a universal modeling standard. While it did not reach that status, it is still supported by a number of rendering programs and web browser plug-ins, many of which are freeware — although some require licensing for certain types of commercial use. The example screen-shots shown in figures 13 and 14 were produced with the Myriad 3D Viewer (Informative Graphics Corporation, `http://www.myriadviewer.com`). A useful feature of this application for checking geometry is its ability to easily display the distance between nodes (panel corner points) on the model, figure 14.

**Figure 13.** Myriad 3D Viewer screen shot produced with VRML 1 output, illustrating hidden-line wire frame rendering on the default black background. Panels are subdivided into triangles for the shading option in the viewer.

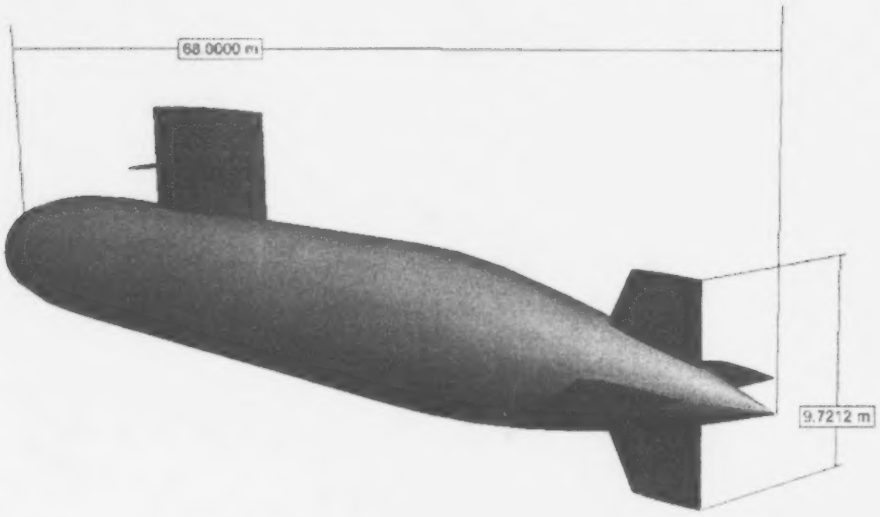


**Figure 14.** Myriad 3D Viewer screen shot produced with VRML 1 output, illustrating shaded rendering with measurements.

## 17 Root.sim Control Deflection Commands

Control deflections are commanded in *Root*.`sim` by the keyword `Delta` or its synonym `Del`. They may also be controlled by one or more autopilots as described in section 10. With geometrical components, the program internally handles each control appendage as an independent entity; this is different from the way that the operator commands deflection of the rudders, sternplanes, etc., and may be different again from how a heading or depth control autopilot is implemented. In order to accommodate these differences, several control deflection command modes are available.

The three principal modes of command can be summarized as:

- direct input, which allows the user to specify an individual appendage to be deflected,
- quasi-direct input, which indicates the type(s) of appendages and implied sense of their deflection, and
- indirect input, which indicates the desired response of the vehicle and equivalent magnitude of the deflection.

There are also a few global control input commands.

Lumps can have the attributes of sternplane, bowplane, and rudder control; these are derived from corresponding coefficients in the Gertler and Hagen equations of motion [19] as presented in annex B. Lumps respond only to quasi-direct and indirect input. Where appropriate, the rules described below for mixing commands apply to lumps also.

Control deflections in a static calculation and commanded deflections in a manoeuvering simulation remain set until they are changed by another `Delta` command. Under autopilot control in a manoeuvering simulation, on the other hand, the commanded deflections, which are in indirect mode, are dynamic. To balance these different requirements, the modes of deflection command are limited in how they work together. In addition, indirect input sets flags that allow only continued indirect input to the affected appendages until canceled by a `Clear` command. This has its origin in requirements for autopilot control: a direct or quasi-direct command is not permitted to 'kick-out' an autopilot. In general, multiple deflection commands to the same appendage are not permitted; however, some combinations of multiple indirect commands are allowed. Command conflicts are discussed in section 17.5.

## 17.1 Direct Deflection Input

| | | |
|---|---|---|
| `Del` | *label*, $\delta$ or *keyword* | or |
| `Del` | $N_L$, $\delta$ or *keyword* | or |
| `Delta` | *label*, $\delta$ or *keyword* | or |
| `Delta` | $N_L$, $\delta$ or *keyword* | |

| Argument(s) | What | Units | Range | Default |
|---|---|---|---|---|
| $N_L$ | index of a lifting appendage | – | (valid) | |
| *label* | label of a lifting appendage | | (valid) | |
| $\delta$ | deflection in local coordinates | deg | | |
| *keyword* | `Clear`, `Reset`, or `Zero` | | | |

The command is directed at an appendage indicated by $N_L$ or *label*; for these to be valid the appendage must have been defined on *Root*`.geo`.

Attention must be given to the sign of $\delta$ because of the local coordinate convention (section 2.2). For example, to deflect horizontal sternplanes to generate upward force, the starboard plane requires positive $\delta$ and the port plane requires negative.

`Clear`: the control surface is returned to its initial deflection and control flags are cleared. The primary use of this command is to cancel indirect input. Unlike `Reset`, below, `Clear` can be combined with another control deflection command: regardless of order, `Clear` is executed first, then the other command.

**Reset**: the control surface is returned to its initial deflection and control flags are cleared. **Reset** cannot be combined with another control deflection command.

**Zero**: the control surface is set at zero deflection and control flags are cleared. **Zero** cannot be combined with another control deflection command.

## 17.2 Quasi-Direct Deflection Input

| **Del** *keyword*, $\delta$ or<br>**Delta** *keyword*, $\delta$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *keyword* | **Bowplane**, **Foreplane**, **Rudder**, **Sailplane**, or **Sternplane** | | | |
| $\delta$ | deflection, see note below | deg | | |

The command is applied to one or more appendage types that are individually associated with a direction of the resulting control force, and also to lumps. The notation $\pm\delta$ used below indicates that the sign of a control deflection depends on that direction; its magnitude is $|\delta|$. (The resultant force direction is indicated for positive $\delta$; it will be opposite for negative $\delta$.)

**Bowplane**: appendages with type **LBowplane** and **SBowplane** are deflected by $\pm\delta$ to produce an upward force. Lumps respond similarly if bowplane coefficients are nonzero.

**Foreplane**: appendages with type **LBowplane**, **SBowplane**, and **Sailplane** are deflected by $\pm\delta$ to produce an upward force. Lumps treat this as a bowplane command.

**Rudder**: appendages with type **Rudder** are deflected by $\pm\delta$ to produce a force to starboard. Lumps respond similarly if rudder coefficients are nonzero.

**Sailplane**: appendages with type **Sailplane** are deflected by $\pm\delta$ to produce an upward force. Lumps treat this as a bowplane command.

**Sternplane**: appendages with type **Sternplane** are deflected by $\pm\delta$ to produce an upward force. Lumps respond similarly if sternplane coefficients are nonzero.

Quasi-direct input is appropriate for a cruciform arrangement of the aft appendages, and for horizontal foreplanes. X-rudders and other non-cruciform arrangements should have the type **Tail**, which does not respond to quasi-direct deflections. Since non-cruciform arrangments are not explicitly represented in lumps, it is assumed that these cruciform-derived commands are interpreted appropriately for them whatever the arrangement.

## 17.3 Indirect Deflection Input

| **Del** *keyword*, $\delta$ or<br>**Delta** *keyword*, $\delta$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *keyword* | **Down**, **Port**, **RollNeg**, **RollPos**, **Stbd**, **Starboard**, or **Up** | | | |
| $\delta$ | deflection, see note below | deg | | |

Indirect commands affect all control appendages that can contribute to the vehicle response indicated by the keyword. Lumps respond similarly. The magnitude of the deflection is weighted individually for each appendage according to $W_{ID}$, $W_{IH}$, $W_{IR}$ introduced in section 3.4, and takes account of its anhedral $k_\phi$, defined below. For both appendages and lumps, vertical plane control deflections are weighted for plane reversal compensation ($C_{pr}$) if speed-gain data is provided, sections 6.3 and 7.3 (automatically for autopilot control, but requiring a **PRevMan** record otherwise). The notation $\pm\ldots\delta$ indicates a deflection of $|\ldots\delta|$ with the appropriate sign.

**Down**: appendages with type **LBowplane**, **SBowplane**, and **Sailplane** are deflected by $\pm W_{ID} k_\phi C_{pr} \delta$ to produce a downward force, and appendages with type **Sternplane** and **Tail** are deflected by $\pm W_{ID} k_\phi C_{pr} \delta$ to produce an upward force. Corresponding lump bowplane and sternplane control deflections are $\pm C_{pr}\delta$.

**Port**: appendages with type **Rudder** and **Tail** are deflected by $\pm W_{IH} k_\phi \delta$ to produce a force to starboard. Corresponding lump rudder deflections are $\pm\delta$.

**RollNeg**: appendages with type **Tail** are deflected by $\pm W_{IR}\delta$ to produce a negative roll moment (i.e., counterclockwise looking forward). Lumps do not respond.

**RollPos**: appendages with type **Tail** are deflected by $\pm W_{IR}\delta$ to produce a positive roll moment (i.e., clockwise looking forward). Lumps do not respond.

**Stbd** or **Starboard**: appendages with type **Rudder** and **Tail** are deflected by $\pm W_{IH} k_\phi \delta$ to produce a force to port. Lump rudder deflections are $\pm\delta$.

**Up**: appendages with type **LBowplane**, **SBowplane**, and **Sailplane** are deflected by $\pm W_{ID} k_\phi C_{pr} \delta$ to produce an upward force, and appendages with type **Sternplane** and **Tail** are deflected by $\pm W_{ID} k_\phi C_{pr} \delta$ to produce a downward force. Corresponding lump bowplane and sternplane control deflections are $\pm C_{pr}\delta$.

The vehicle response indicated is for a positive value of $\delta$; the response is opposite if $\delta$ is negative. Therefore, '**Up +10**' produces the same result as '**Down -10**'. For **Up**, **Down**, **Port**, and **Stbd**, appendage deflections are multiplied by anhedral factor $k_\phi = \cos\phi$, where $\phi$ is appendage anhedral in the appropriate crossflow direction (horizontal or vertical).

Indirect deflection commands are processed independently according to whether they are associated with depth control (`Up` and `Down`), heading control (`Port` and `Stbd`), or roll control (`RollNeg` and `RollPos`). Although multiple input in any one mode is not permitted, mixed indirect input is allowed.

For X-rudders with type `Tail`, the following mixed input will result in the rudders being deflected so as to produce both responses simultaneously.

```
Del Stbd 10
Del Down  5
```

Note that an appendage under indirect control requires a `Clear` command before it can be given a direct or quasi-direct deflection command. This applies to lumps also, but they cannot be cleared individually: an '`All Clear`' command, section 17.4, is required for them.

## 17.4 Global Deflection Input

| **`Del`** **`All`**, *keyword* or<br>**`Delta`** **`All`**, *keyword* | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| `All` | indicates global input | | | |
| *keyword* | `Clear`, `Reset`, or `Zero` | | | |

`Clear`: all control surfaces and lump control modes are returned to their initial deflection and control flags are cleared. The primary use of this command is to cancel indirect input. (However it does not cancel autopilot control; attempting that results in an error. Autopilots must be explicitly `Stopped`, section 26.2.) Unlike `Reset`, below, `Clear` can be combined with other control deflection commands; regardless of order, the global `Clear` is executed first, then the other commands.

`Reset`: all control surfaces and lump control modes are returned to their initial deflection and control flags are cleared. `Reset` cannot be combined with any other control deflection commands.

`Zero`: all control surfaces and lump control modes are set at zero deflection and control flags are cleared. `Zero` cannot be combined with any other control deflection commands.

## 17.5 Conflicting Deflection Commands

`DSSP50` tries to prevent deflection command conflicts. Between static simulations, or simultaneously for dynamic simulations, only the following combinations of deflection commands are allowed for any individual appendage:

- a direct command with global `Clear`,
- a quasi-direct command with global `Clear`,
- an indirect-depth, indirect-heading, and indirect-roll command with each other, and with global `Clear`,
- global `Reset` or `Zero`.

Lump control modes respond analogously. Only the following are allowed for each lump control mode::

- a quasi-direct command with global `Clear`,
- an indirect-depth and indirect-heading command with each other, and with global `Clear`,
- global `Reset` or `Zero`.

A fatal error is generated for any other combination.

Commanded deflections stay in effect until legitimately canceled or replaced by a new command. An appendage under indirect control requires a `Clear` command before or coincident with returning it to direct or quasi-direct control. Failure to do so generates a fatal error. In a free manoeuvering simulation, active depth, heading, and roll autopilots use indirect deflection control, and the same rule applies. Issuing a `Stop` command to an active autopilot cancels the indirect commands associated with it.

## 18 Root.sim Heading Specification

The specifications discussed in this section are used in a free manoeuver to define initial compass heading and to interpret commands to the heading autopilot. Specifically, they constitute *string* in the following records:

| | | |
|---|---|---|
| `AutoHead` | [*keyword*], [*keyword*], [*string*] | and |
| `Heading` | *string* | |

which are described in sections 26.1 and 26.2. The *string* consists of a numerical value for heading combined with up to two keywords. In what follows, it may be useful to refer to figure 2. Note that heading angle increases in a turn to starboard.

Within `DSSP50`, instantaneous heading is the azimuthal angle $\psi$ relative to the initial heading (i.e., $\psi_\bullet = 0$). In a free manoeuver, this is tabulated on `sim`*nnn*`.txt` together with the equivalent compass heading based on an initial compass heading input by the user. By default this initial heading is zero degrees, or North. The *string* format allows heading to be specified relative to the initial reference, relative to the current heading, or with respect to the compass.

It is easiest to simply list the different formats, a to h. Heading is evaluated from *string* as follows, where *value* is numerical:

a. *value* : azimuthal degrees relative to the initial reference.

b. *value* `Points` : azimuthal angle in points relative to the current heading.

c. *value* `Port/Stbd/Starboard` : degrees to port or starboard of the current heading.

d. *value* `Points Port/Stbd/Starboard` : points to port or starboard of the current heading.

e. *value* `Relative` : degrees relative to the current heading.

f. *value* `Azimuth` : degrees azimuth on the compass.

g. `N/North/NbyE/.../NbyW` : compass heading. See annex A for the valid keywords; they may not be abbreviated for input.

h. `North` *value* `East`, `N` *value* `W`, etc. : compass quadrant heading in degrees. The keywords may not be abbreviated for input.

`DSSP50` outputs current heading in the range $-180° \leq \psi \leq 180°$ if quaternions are used to represent the Euler angles (section 28), but heading is unbounded otherwise. Compass heading is always output in the range 0 to 360 degrees.

The following examples illustrate commanded (subscript $c$) heading resulting from the various *string* formats. It is assumed that the initial compass heading in the manoeuver was `East`, and that the current heading $\psi$ is 20 degrees (i.e., a compass heading of 110 degrees).

| Format | *string* | $\psi_c$ | $\text{Compass}_c$ |
|---|---|---|---|
| a. | −25 | −25.00° | 65.00° |
| b. | 2 Points | 42.50° | 132.50° |
| c. | 15 Port | 5.00° | 95.00° |
| d. | 3 Points Port | −13.75° | 76.25° |
| e. | −25 Relative | −5.00° | 85.00° |
| f. | −25 Azimuth | −115.00° | 335.00° |
| g. | NWbyW | −146.25° | 303.75° |
| h. | S 40 E | 50.00° | 140.00° |

While a great deal of flexibility is provided in these options, it should be obvious that mixing commands based on current heading and compass heading in the same input file is a potential source of confusion, and users should exercise due care.

The compass heading and the current heading default to zero. Therefore, for the purpose of inputting a specific initial heading, formats a, e, and f are exactly equivalent to each other, and the values in each of formats a to f represent the compass heading relative to zero (North).

# 19 Root.sim MBT Blowing and Venting Commands

Ballast blowing is commanded during a free manoeuver by the keyword **Blow**, section 26.2. (Like others, the same command is issued internally in a MLD simulation, but that is transparent to the user.)

| **Blow** *argument 1*, [*argument 2*] | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *argument 1* | identifies the MBT(s) to be blown | | | |
| *argument 2* | identifies the BG(s) to be used | | | **All** |

The first argument is one of:

$N_B$: the index of the MBT to be blown (the $N_B{}^{th}$ tank defined on *Root*.**geo** has the index $N_B$).

*label*: the label of the MBT to be blown.

**Aft/Forward/Fwd**: all MBTs of the type indicated are to be blown.

**All**: all MBTs are to be blown.

**Stop**: stop blowing all MBTs.

And the second argument is one of:

$N_G$: the index of the BG to be used (the $N_G{}^{th}$ bottle group defined on *Root*.**geo** has the index $N_G$).

*label*: the label of the BG to be used.

**Aft/Forward/Fwd/Main**: all BGs of the type indicated are to be used.

**All**: all BGs associated with the tank(s) identified by *argument 1* are to be used.

**Stop**: stop blowing the tank(s) identified by *argument 1*.

The following examples illustrate how these arguments are combined:

| | |
|---|---|
| **Blow 1** | blow MBT 1 with all associated BGs |
| **Blow 2 Fwd** | blow MBT 2 with BGs of type **Forward**, if associated |
| **Blow All Main** | blow all MBTs with **Main** BGs |
| **Blow Aft** | blow all MBTs of type **Aft** with **Aft** BGs |
| **Blow 3 Stop** | stop blowing MBT 3 |
| **Blow Stop** | is the same as **Blow All Stop** |

Care has to be exercised with blowing commands because BGs and MBTs are related only by simple association. If there is any doubt what the associations are, refer to the table that is output on *Root*.**spr** for a free or MLD simulation. In addition there are ambiguities: in section 13 it was noted that a bottle group may be connected so that it can only blow two tanks together, as is BG1 in figure 15, although a simple

association does not reflect that. Therefore, we would represent the system in figure 15 the same way as for figure 8: by giving BG1, MBT1, and MBT2 type `Forward`, giving BG2 type `Main`, and giving BG3 and MBT3 type `Aft`. Blowing MBT1 and MBT2 simultaneously from just BG1 is achieved with '`Blow Fwd Fwd`' (or '`Blow All Fwd`'). However, there is nothing to prevent the user from using '`Blow 1`' or other commands that may inadvertently result in BG1 blowing MBT1 alone, which is consistent with figure 8 but not with figure 15.

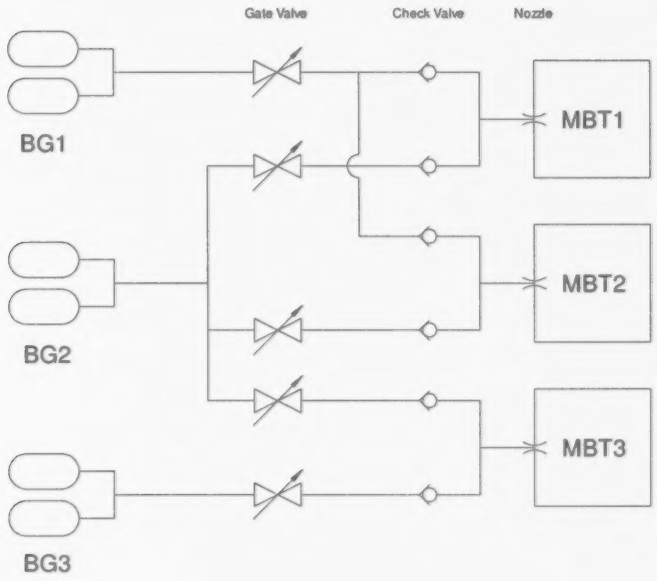


**Figure 15.** A variation on figure 8: BG1 can only blow MBT1 and MBT2 together, BG2 can blow any of the three individually or together, and BG3 can blow only MBT3.

The present build of the program allows two types of ballast blows: isothermal, which is a common approximation in simulation codes, and adiabatic, the default; they are discussed in more detail in reference [2]. The author is not aware of any reliable data for validation of either method, but it is believed that an isothermal blow overestimates the blowing rate, whereas the adiabatic blow as implemented in DSSP50 likely underestimates it. Adiabatic blowing is the default because it appears closest to reality while being conservative with respect to the effectiveness of the blow. Isothermal blowing is initiated by the keyword `Isothermal`.

The command to vent a ballast tank is simpler. It is assumed that air is vented at the rate set in MBT level 2 input, section 13.2; this rate is constant and independent of depth. There is no heat transfer associated with venting.

| `Vent` *argument 1*, [`Stop`] | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *argument 1* | identifies the MBT(s) to be vented | | | |
| `Stop` | stop venting the identified MBT(s) | | | |

The first argument is one of:

$N_B$: the index of the MBT to be vented (the $N_B{}^{th}$ tank defined on *Root*.`geo` has the index $N_B$).

*label*: the label of the MBT to be vented.

`All`: all MBTs are to be vented.

`Stop`: stop venting all MBTs.

## 20 Root.sim WTC Flooding Commands

A flood in a watertight compartment (WTC) is initiated during a free manoeuver by the keyword `Flood` (section 26.2).

| `Flood` *argument 1*, [*argument 2*, [$C_F$]] | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| *argument 1* | identifies the WTC to be flooded | | | |
| *argument 2* | breach diameter or `Stop` | | | |
| $C_F$ | coefficient of discharge | – | $0 \leq \ldots \leq 1$ | 0.6 |

The first argument is one of:

$N_C$: the index of the WTC to be flooded (the $N_C{}^{th}$ WTC defined on *Root*.`geo` has the index $N_C$).

*label*: the label of the WTC to be flooded.

`Stop`: secure floods in all WTCs.

And the second argument is one of:

$D_F$: the diameter of the breach, which is assumed to be circular (otherwise, use the diameter of a circular hole with the same area as the breach).

`Stop`: secure the flood in the WTC identified by *argument 1*.

If the first argument is a WTC index or label then the second argument is mandatory. The third argument is only used if the second is $D_F$. Unless the user has information about the value to use for a specific system breach, $C_F$ can be omitted; the default coefficient of discharge will give acceptable results for most purposes.

The following examples illustrate how the arguments are combined:

| | |
|---|---|
| `Flood Stop` | secure all floods in progress |
| `Flood 2 0.08 0.45` | flood in WTC 2 with $D_F = 0.08$ and $C_F = 0.45$ |
| `Flood #ControlRoom Stop` | secure flood in labeled WTC |

Note that '`Flood 1 0`' is the same as '`Flood 1 Stop`'.

If a new flood is started in a watertight compartment where one is already in progress, flooding continues with the new parameters $D_F$ and $C_F$. Therefore, to represent an additional flood in the same WTC, the value of $D_F$ should correspond to the combined area of the existing and new breaches.

## 21 Root.sim Trim and Compensation Commands

A schematic trim and compensation system is shown in figure 16. Interlocks to prevent simultaneous trim and compensation action on a TC tank are not shown; neither are tank venting arrangements.

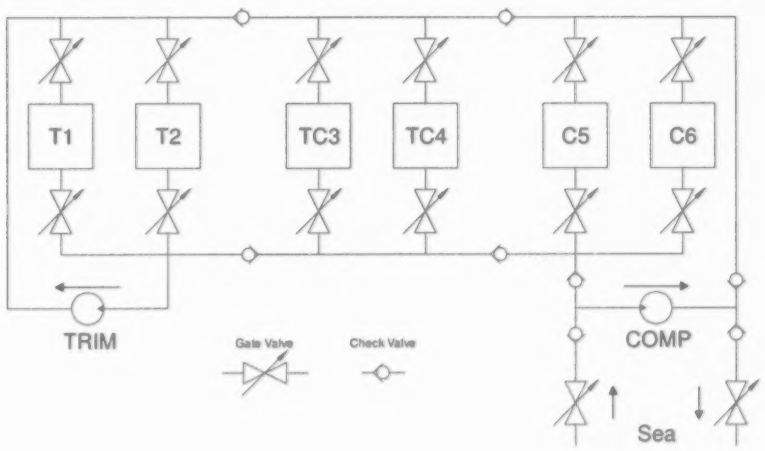


**Figure 16.** Schematic of a trim and compensation system with two `Comp` (C) tanks, two `Trim` (T) tanks, and two `TrimComp` (TC) tanks. All fill at the top and empty at the bottom.

The distinction between `Comp` and `Trim` tanks permits the program to disallow inappropriate commands to the system. For simplification, all tanks could be defined as `TrimComp`, but then the user must ensure that only appropriate commands are issued.

Tanks that are free-flooded and blown with HP air, such as Q, safety, and negative tanks, are better treated as MBTs rather than as part of the trim and compensation system.

Trim and compensation are only used in a **Free** simulation. The initial occupancy of a tank of any type is set with the record:

| **TrimComp** $N_{tct}$, $V_{Otct}$ or<br>**TrimComp** *label*, $V_{Otct}$ | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $N_{tct}$ | index of the tank in question | – | | |
| *label* | label of the tank in question | | | |
| $V_{Otct}$ | initial occupied volume of the tank | $m^3$ | $0 \leq \ldots \leq V_{tct}$ | $V_{tct}/2$ |

$V_{tct}$ is the total volume of the tank. Clearly, it makes no sense for all **Trim** tanks to be initially empty unless there are also **TrimComp** tanks (which can take on seawater for trimming purposes) present.

The commands to control compensation and trim during a **Free** simulation are:

| **Comp** $m_t$, *keyword*, $N_{tct}$ or *label* or<br>**Comp Stop** | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $m_t$ | mass of water to be transferred | kg | $0 < \ldots$ | |
| *keyword* | **IN** or **OUT** | | | |
| $N_{tct}$ | index of the tank in question | – | | |
| *label* | label of the tank in question | | | |
| **Stop** | stops any current compensation action | | | |

**IN** or **OUT** respectively cause the compensation system to take in or expel the indicated amount of seawater to or from the tank in question. If it is a **Trim** tank, or if it is a **TrimComp** tank that is active in a **Trim** transfer, an error is generated.

| **Trim** $m_t$, $N_{tct}$ or *label*, $N_{tct}$ or *label* or<br>**Trim Stop** | | | | |
|---|---|---|---|---|
| Argument(s) | What | Units | Range | Default |
| $m_t$ | mass of water to be transferred | kg | $0 < \ldots$ | |
| $N_{tct}$ | index of the first or second tank | – | | |
| *label* | label of the first or second tank | | | |
| **Stop** | stops any current trimming action | | | |

The trim system will transfer the indicated amount of seawater from the first tank to the second tank. If either is a `Comp` tank, or is a `TrimComp` tank that is active in a `Comp` transfer, or if they are identical, an error is generated.

Because it is assumed that there is only one pump for each function, both `Comp` and `Trim` cancel another transfer of the same type already in progress. The following examples illustrate compensation and trim commands (a preceding value of time is assumed):

| | |
|---|---|
| `Comp 500 Out 3` | expel 0.5 t from `Comp` (or `TrimComp`) tank 3 |
| `Trim 1000 #FwdTank #AftTank` | transfer 1 t aft (i.e., between labelled tanks) |

The action occurs at the appropriate rate until either the required amount is transferred or a limit is reached by a tank being full or empty. Details including transfer time are output to *Root*`.log`.

# 22 Root.sim Level 1 Input

*Root*`.sim` level 1 input defines a sequence of calculations or simulations to be run. Each simulation is defined by a block of level 2 input records, and it is executed before control is passed back to level 1. There may be up to 999 simulations in the sequence.

The *Root*`.sim` keyword records at level 1 are:

| Keyword | Function |
|---|---|
| `Captive` | run captive vehicle manoeuvering load calculations |
| `Coef` | generate a set of hydrodynamic coefficients |
| `Coefficnt` | generate a set of hydrodynamic coefficients |
| `Free` | run a free-swimming vehicle simulation |
| `Help` | send help output to *Root*`.log` |
| `MLD` | generate Manoeuvering Limitation Diagram curves |
| `Output` | define output control parameter |
| `Static` | run a set of static load calculations |
| `Text` | title or other information, see section 3.4 |
| `Warnings` | define warning control parameter |

## 22.1 Calculation and Simulation Specifications

These records introduce level 2 input for a simulation. Execution of each one adds to *Root*`.spr` output and generates a new `sim`*nnn*`.txt` output file. `Free` and `MLD` simulations also add a new `aux`*nnn*`.txt` file.